BYZANTINE-RESILIENT DECENTRALIZED LEARNING

by

ZHIXIONG YANG

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Waheed U. Bajwa

and approved by

New Brunswick, New Jersey

January, 2020

ABSTRACT OF THE DISSERTATION

Byzantine-resilient Decentralized Learning

By Zhixiong Yang

Dissertation Director:

Waheed U. Bajwa

When datasets are distributed over a network and a central server is infeasible, machine learning has to be performed in a decentralized fashion. The dissertation introduces new methods that solve decentralized machine learning problems in the presence of Byzantine failures. Classic decentralized learning methods require nodes communicate with each other by communicating over the network. When a node engages in arbitrary or malicious behavior, it is termed as having Byzantine failure. Without any Byzantine-resilient modification, classic learning methods cannot complete machine learning tasks as intended in the presence of Byzantine failure. Byzantine-resilient decentralized learning methods are discussed in this dissertation. Both theoretical guarantees and experiments are given to justify the usefulness of the methods under Byzantine settings.

Table of Contents

Chapter 1

Introduction

In the view of machine learning, data has some probability distribution. Machine learning techniques complete tasks by applying a risk function on the data and statistically minimizing the risk function. What makes the machine learning problem challenging is that the data distribution is usually unknown. One of the ways to tackle this issue is to employ empirical risk minimization (ERM). ERM usually requires a dataset drawn from the data distribution. Then, instead of directly minimizing the statistical risk function, ERM minimizes the empirical risk function on the dataset. It is shown that, as the size of dataset increases, both the minimum and minimizer of the empirical risk function converge to the minimum and minimizer of the statistical risk, respectively [1–4].

In real applications, how the dataset is stored and processed makes a difference both in the design and performance of algorithms. When the dataset is available and can be processed at a single location, we call this machine learning problem **centralized learning**. In some applications, multiple datasets may be distributed over a network that is constructed by connecting **nodes** with edges. Datasets in this case are usually stored at each node. More often than not, it is inefficient to gather all the datasets to a centralized location and perform centralized learning. To overcome this difficulty, we need help from **distributed learning** and **decentralized learning**. In the scenario when there is a **central server** directly connected to all the nodes, we call this network structure a **distributed network**. On the contrary, when there is no central server in the network and each node can only communicate with a subset of nodes, we call this network structure a **decentralized network**. With the advanced development of data science, communication, computing, and machine learning techniques, distributed and decentralized learning have been extensively studied in recent years [5–8].

Since learning over a network requires nodes to cooperate with each other, most of the existing works on learning over the network are developed under the assumption that all the nodes are cooperative. However, in the real world, an individual node is vulnerable to failures and attacks. Without proper compensation of such threats, the existence of non-functional or malicious nodes can easily jeopardize the functionality of the whole network. One good model of the failures and malicious attacks is called **Byzantine failure** [9, 10]. Byzantine failure, meaning a node can arbitrarily deviates from its intended behavior, is generally considered the type of failure that is hardest to safeguard against. A Byzantine node can potentially inject false information into the network, collaborate with other Byzantine nodes, and pretend to behave normally in order not to get caught. While Byzantine-resilient distributed learning algorithms have been attracting attention recently [11–16], the area of decentralized Byzantine-resilient machine learning is relatively open. Our study in this dissertation is focused on Byzantine-resilient machine learning.

1.1 Machine learning over networks

While the dissertation is focusing on decentralized machine learning, the first question we need to address is: why not use centralized learning? Or in other words, when is learning over networks preferred over centralized learning? One argument in favor of centralized learning could be that dealing with all the data at a centralized location is simpler: no need to worry about the communications; easier to design algorithms; better privacy in the sense that no information needs to be shared with others. However, with fast development of technologies, more parameters need to be considered to deal with more complicated situations. One example could be dealing with big data. It is claimed that over 2.5 quintillion bytes of data are created every single day [17]. In the big data scenario, the huge amount of data cannot be processed by a single computing unit. It is much more efficient if one can distribute the data onto multiple nodes and improve the computing capability by letting the computing units cooperate over a network. In some other cases, data is generated in a distributed nature. For applications like the Internet of Things (IoT), data-generating devices are connected over a network. Instead of transferring raw data to a centralized location, processing data on multiple devices in a distributed or decentralized fashion brings the advantage of high efficiency, low communication cost and real-time response. Another aspect could be privacy concerns. Sensitive data such as personal data on a smart phone should not be shared with others. Learning over the network is a privacy-preserving way to learn from the private data without actually having access to such data.

While a number of Byzantine-resilient learning algorithms will be discussed in this dissertation, none of these method requires any node to transfer any data or gain access to datasets other than the dataset on the node itself. Since there are multiple nodes in the network, the dataset stored and processed on each node is considerably smaller than the total size of the data within the network. So learning over the network can potentially require less storage capacity and computing power on each node, achieve faster processing speed, and preserve privacy. Additionally, as we are going to show in the dissertation both theoretically and numerically, competitive models can be learned over the network without having more data than centralized learning methods.

1.2 Byzantine failure

While learning over the network brings advantages in applications, it is also generally a more complicated problem than centralized learning. One of the difficulties lies in the robustness of the learning algorithms. When having a large number of nodes in the network, it is hard to guarantee that all nodes are working as intended. Therefore, dealing with failures in the network becomes one of the necessary and practical consideration for machine learning tasks. In the literature, failures that happen during learning over the network can be modeled and categorized into crash failure, omission failure, timing failure and Byzantine failure.

Byzantine failure model, meaning that a node can arbitrarily deviate from its intended behavior, was originally brought up in [9] and abstracted as the Byzantine Generals Problem. The article [9] describes the scenario where a number of Byzantine generals are surrounding a city and trying to reach an agreement on the attacking strategy. When there are traitors among the generals, but the traitors cannot be identified based on their proposed strategy, a good decision rule has to be enforced to guarantee that an agreement can be achieved and that a small number of traitors cannot overturn the intended strategy. This is also why the failure model is called Byzantine failure. Among the aforementioned failure types, Byzantine failure is the hardest to safeguard against in the sense that we do not put any assumption on its behavior pattern. A Byzantine node can potentially be aware of the network structure, collude with other Byzantine nodes, acquire knowledge of messages transmitting in the network, inject harmful information into the network, pretend to cooperate to avoid being caught and so on and so forth.

In this dissertation, we discuss machine learning algorithms that are Byzantine-resilient, meaning that the algorithms can accomplish machine learning tasks despite the existence of Byzantine failures. In the introduced methods, instead of trying to identify Byzantine nodes, the algorithms are designed to tolerate a certain number of Byzantine nodes in the network. We then show that the proposed algorithms can successfully learn a good model if there are not too many Byzantine nodes in the network.

1.3 Decentralized learning

As mentioned previously, machine learning methods can be categorized as distributed learning (with central server) and decentralized learning (without central server). In this dissertation, we mainly focus on the discussion of Byzantine-resilient decentralized learning methods.

Although the datasets are distributed on the nodes over a network in a similar fashion for both distributed and decentralized settings, the nature of the two types methods are fundamentally different. Because the central server is directly connected to all the nodes, it is possible to broadcast the model to all the nodes so that all nodes can share the same model during each iteration. Distributed learning algorithms take this advantage and perform gradient aggregation as the following: each node computes the gradient with respect to the same model; then the server collects all the gradients and updates the model according to some aggregation rule.

The case for decentralized learning is more complicated. Without a central server, all nodes have to rely on the algorithm to iteratively achieve consensus, meaning that all nodes eventually agree on the same model. Unfortunately, before consensus is reached, gradient aggregation cannot be applied since the gradient computed at one node cannot be used to update the different model on another node. Thus existing decentralized learning works mostly focus on model aggregation, meaning that each node shares its local model with other nodes. Then each node updates its local model based on both the computation on local dataset and the models received from other nodes. As a result, the strategy to safeguard against Byzantine failures in the two settings are also fundamentally different.

In most decentralized applications, each node can only directly communicate with a fraction of nodes, which are called neighbors. The connectivity of the nodes and their neighbors can be modeled as an incomplete graph. Intuitively, the connectivity of the graph should also have impact on both the convergence rate and the robustness of the algorithms. Thus, different from distributed algorithms, decentralized algorithms also need to address the influence on different network topology. In other words, in order for certain algorithms to work, the network needs to satisfy some topology constraints. It is then easy to see that the strategy of rejecting Byzantine failures will also consider the network topology. Generally speaking, if the total number of nodes stays the same, a decentralized network can tolerate fewer Byzantine nodes than a distributed network.

Although learning over a decentralized network is more complicated than under distributed settings, it also has advantages over distributed learning. One limitation of distributed network is the requirement of having a central server that is connected to all other nodes. In applications, connecting to all nodes is not always feasible or can be too costly. If there are a large number of nodes in the network, the communication load on the server is heavy and likely to become the bottleneck [18]. Since decentralized algorithms only require each node to be able to communicate with its neighbors, the number of nodes can scale well without requiring more communication capability. In terms of Byzantine failure, it usually takes a large enough fraction of nodes to become Byzantine nodes to successfully crash the whole decentralized network, while in the distributed network, if the central server is taken over, the whole network is paralyzed.

1.4 Problem formulation

Given a network in which each node has access to some local training data, the main goal of this dissertation is to develop algorithms that learn models from these data in a decentralized fashion when there are Byzantine failures in the network. We are also interested in both the statistical and algorithmic convergence rate.

1.4.1 Learning model

We consider a network of M nodes, expressed as a directed, static graph $\mathcal{G}(\mathcal{J}, \mathcal{E})$. Here, the set $\mathcal{J} := \{1, \dots, M\}$ represents nodes in the network, while the set of edges $\mathcal{E}$ represents communication links between different nodes. Specifically, $(j, i) \in \mathcal{E}$ if and only if node i can receive information from node j and vice versa. Each node j has access only to a local training set $\mathcal{Z}_j = \{\mathbf{z}_{jn}\}_{n=1}^{|\mathcal{Z}_j|}$. Training samples $\mathbf{z}$, are assumed independent and identically distributed (i.i.d.) and drawn from an unknown distribution $\mathcal{P}$, i.e., $\mathbf{z}_{jn} \sim \mathcal{P}$. For simplicity, we assume cardinalities of local training sets are the same, i.e., $|\mathcal{Z}_j| = N$. The generalization to the case when $\mathcal{Z}_j$'s are not equal sized is trivial.

Machine learning tasks are usually accomplished by defining and statistically minimizing a risk function $\mathbb{E}_{\mathbf{z}\sim\mathcal{P}}[f(\mathbf{w}, \mathbf{z})]$ with respect to a variable $\mathbf{w} \in \mathbb{R}^d$. For simplicity, we use $\mathbb{E}[f(\mathbf{w})]$ in the following to denote the statistical risk function and $\nabla f(\mathbf{w})$ to denote the gradient of $f(\mathbf{w}, \mathbf{z})$ with respect to $\mathbf{w}$ in this dissertation. We denote the true minimizer of the risk function as $\mathbf{w}^*$, i.e.,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}[f(\mathbf{w})]. \tag{1.1}$$

In learning problems, the distribution $\mathcal{P}$ is usually unknown. Therefore $\mathbf{w}^*$ cannot be solved for directly. One way of completing the task in this (decentralized) setting is to employ (decentralized) empirical risk minimization (ERM), i.e.,

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{MN}\sum_{j=1}^{M}\sum_{n=1}^{N} f(\mathbf{w}, \mathbf{z}_{jn}) \triangleq \min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{M}\sum_{j=1}^{M} f_j(\mathbf{w}). \tag{1.2}$$

It is shown that the minimizer of (1.2) converges to $\mathbf{w}^*$ with high probability as MN increases [19].

In decentralized learning, each node j maintains a local variable $\mathbf{w}_j$. The ERM problem can be solved in the decentralized fashion [20], i.e.,

$$\min_{\{\mathbf{w}_1,\ldots,\mathbf{w}_M\}} \frac{1}{M}\sum_{j=1}^{M} f_j(\mathbf{w}_j) \ \text{ subject to } \ \mathbf{w}_i = \mathbf{w}_j \ \forall i, j. \tag{1.3}$$

To accomplish the decentralized ERM task, all nodes need to cooperate with each other by communicating with neighbors over edges. We define the neighborhood of node j as $\mathcal{N}_j := \{i \in \mathcal{J} : (i,j) \in \mathcal{E}\}$. If $i \in \mathcal{N}_j$, then i is a neighbor of node j. Classic decentralized learning algorithms proceed iteratively. A node is expected to accomplish two tasks during each iteration: update the local variable $\mathbf{w}_j$ according to some rule $g_j(\cdot)$ and broadcast a message to all its neighbors. Note that node j can receive values from node i only if $i \in \mathcal{N}_j$.

1.4.2 Byzantine failure model

When there is no failure in the network, decentralized learning is well understood [21, 22]. The main assumption in this dissertation is that some of the nodes in the network can arbitrarily deviate from intended behavior. We model this behavior as Byzantine failure, formally defined as follows.

Definition 1. *A node $j \in \mathcal{J}$ is said to be Byzantine if during any iteration, it either updates its local variable using an update function $\widetilde{g}_j(\cdot) \neq g_j(\cdot)$ or it broadcasts a value other than the intended update to its neighbors.*

We use $\mathcal{J}'$ to denote the set of nonfaulty nodes and we assume that there are at most b Byzantine nodes in the network. We label the nonfaulty nodes from 1 to $|\mathcal{J}'|$ without loss of generality. Here we emphasize that we do not need to know the exact number of Byzantine nodes. A Byzantine-resilient learning algorithms should be able to tolerate at most b Byzantine nodes and the algorithm should still have a competitive performance even if there is actually no failure in the network.

In this dissertation, we will introduce Byzantine fault-tolerant algorithms for decentralized learning. A valid algorithm is expected to accomplish the following tasks: (i) achieve consensus, i.e., $\mathbf{w}_j = \mathbf{w}_i \ \forall i, j \in \mathcal{J}'$ as the number of iterations $t \to \infty$; and (ii) learn a $\mathbf{w}_j \to \mathbf{w}^* \ \forall j \in \mathcal{J}'$ as sample size $N \to \infty$.

The rest of this dissertation is organized as the following. We first introduce a Byzantine-resilient algorithm that is specifically designed for decentralized support vector machine (SVM) in Chapter 2. We then show the possibility for probably and approximately correct (PAC) learning for a wide class of learning problems in the presence of Byzantine failures along with an implementable algorithm in Chapter 3. Next, we present and analyze a general framework for Byzantine-resilient decentralized machine learning in Chapter 4. At last, we provide some discussion on open research problems and conclude the dissertation in Chapter 5.

Chapter 2

Byzantine-resilient Decentralized SVM

One commonly adopted idea of rejecting Byzantine failures is to combine screening methods from robust statistics with decentralized learning methods. Different algorithms can be developed with different combinations of optimization methods and screening methods. In this chapter, we introduce our first work that focuses on a particular type of vector-valued decentralized learning problem in the presence of Byzantine failures, i.e., training of a support vector machine (SVM) [23].

2.1 Byzantine-resilient support vector machine (RD-SVM)

An SVM is a popular tool to solve binary classification problems. In this setting, training sample $\mathbf{z} = (x, y)$, where $x \in \mathbb{R}^P$ is the data and $y \in \{1, -1\}$ is its label. The variable $\mathbf{w}$ in this case takes the form $\mathbf{w} = [\mathbf{a}^T, c]^T$ where $\mathbf{a} \in \mathbb{R}^P$ and $c \in \mathbb{R}$. It is desired to learn a mapping $y = \mathbf{a}^T x + c$ that correctly maps each data sample to its label. The loss function of SVM is hinge loss $\ell(\mathbf{w}, \mathbf{z}) = \max\{0, 1 - y(\mathbf{a}^T x + c)\}$ and the regularizer $R(\mathbf{a}) = \frac{\lambda}{2}\|\mathbf{a}\|^2$. Thus the risk function $f(\mathbf{w}, \mathbf{z}) = R(\mathbf{a}) + \ell(\mathbf{w}, \mathbf{z})$ is strongly convex.

In the decentralized setting, each node j has access to a local training set $(\mathbf{X}_j, \mathbf{Y}_j)$ where $\mathbf{X}_j$ is a $P \times N$ matrix whose columns are training samples x_{jn} and $\mathbf{Y}_j$ is a $N \times N$ diagonal matrix $\text{diag}(\{y_{jn}\})$. Every node j also keeps a local $\mathbf{w}_j = [\mathbf{a}_j^T, c_j]^T$ and can exchange the latest $\mathbf{w}_j$ over the edges. All nodes are expected to agree on the same $\widehat{\mathbf{w}} = [\widehat{\mathbf{a}}^T, \widehat{c}]^T$ that minimizes the global empirical risk function, i.e.,

$$\begin{aligned}(\widehat{\mathbf{a}}, \widehat{c}) = &\arg\min_{\{\mathbf{a}_j, c_j\}} \frac{\lambda}{2}\sum_{j=1}^{M}\|\mathbf{a}_j\|^2 + \sum_{j=1}^{M}\sum_{n=1}^{N}\max\{0, 1 - y_{jn}(\mathbf{a}_j^T x_{jn} + c_j)\} \\ &\text{s.t.} \qquad \mathbf{a}_j = \mathbf{a}_i, \quad c_j = c_i, \quad \forall i, j \in J. \end{aligned} \tag{2.1}$$

Without any failure, the problem can be solved by iteratively proceeding with the following updates [21]:

$$\Lambda_j(t+1) = \underset{\mathbf{0} \preceq \Lambda_j(t) \preceq \mathbf{1}/\lambda}{\arg\max} \; -\frac{1}{2}\Lambda_j(t)^T \mathbf{Y}_j \mathbf{X}_j \mathbf{U}_j^{-1} \mathbf{X}_j^T \mathbf{Y}_j \Lambda_j(t) + (\mathbf{1} + \mathbf{Y}_j \mathbf{X}_j \mathbf{U}^{-1} \xi_j(t))^T \Lambda_j(t) \tag{2.2}$$

$$\mathbf{w}_j(t+1) = \mathbf{U}_j^{-1}(\mathbf{X}_j^T \mathbf{Y}_j \Lambda_j(t+1) - \xi_j(t)) \tag{2.3}$$

$$\zeta_j(t+1) = \zeta_j(t) + \frac{\rho}{2}\sum_{i\in\mathcal{N}_j}(\mathbf{w}_j(t+1) - \mathbf{w}_i(t+1)), \tag{2.4}$$

Where $\preceq$ denotes element-wise comparison. Here, ρ is a pre-selected stepsize, $\mathbf{U}_j = (1+2\rho|\mathcal{N}_j|)\mathbf{I} - \Pi$ and $\xi_j(t) = 2\zeta_j(t) - \rho\sum_{i\in\mathcal{N}_j}(\mathbf{w}_j(t)+\mathbf{w}_i(t))$, where Π is a $(P+1)\times(P+1)$ matrix with zeros everywhere except for $[\Pi]_{(P+1)(P+1)} = 1$. However, when there are Byzantine failures in the network, the given algorithm will fail. For example, if one of the nodes holds its local value invariant for all iterations, because of the equality constraint in (2.1), all nonfaulty nodes will iteratively converge to the same value that the Byzantine node holds.

The key idea to make the algorithm Byzantine resilient is to add a screening step before using the information received from a node's neighbors in (2.4). After node j receives the classifiers from all neighbors in $\mathcal{N}_j$, it computes the loss of these classifiers on its local training set S_j and uses this information to perform the screening. First, separate the local set S_j into a positive set $S_j^p = \{(x,y) \in S_j : y = 1\}$ and a negative set $S_j^n = \{(x,y) \in S_j : y = -1\}$. Compute $\ell_j^p = \sum_{(x,y)\in S_j^p} \max\{0, 1 - y(\mathbf{a}_j^T x + c_j)\}$ and $\ell_j^s = \sum_{(x,y)\in S_j^s} \max\{0, 1 - y(\mathbf{a}_j^T x + c_j)\}$. Then for each $i \in \mathcal{N}_j$, compute $\ell_i^p = \sum_{(x,y)\in S_j^p} \max\{0, 1 - y(\mathbf{a}_i^T x + c_i)\}$ and $\ell_i^s = \sum_{(x,y)\in S_j^s} \max\{0, 1 - y(\mathbf{a}_i^T x + c_i)\}$. Next, formulate a new neighborhood $\mathcal{N}_j^*(t)$ by eliminating a group of nodes in the following way: (1) if there are less than b nodes that satisfy $\ell_i^s \geq \ell_j^s$, eliminate all theses nodes; (2) if there are no less than b nodes that satisfy $\ell_i^s \geq \ell_j^s$, eliminate b nodes that have the b greatest ℓ_j^s; (3) if there are less than b nodes that satisfy $\ell_i^p \geq \ell_j^p$, eliminate all theses nodes; (4) if there are no less than b nodes that satisfy $\ell_i^p \geq \ell_j^p$, eliminate b nodes that have the b greatest ℓ_j^p; (5) collect all the nodes that are not eliminated into $\mathcal{N}_j^*$. The algorithm then can be modified as

$$\Lambda_j(t+1) = \underset{\mathbf{0}\preceq\Lambda_j(t)\preceq\mathbf{1}/\lambda}{\arg\max} -\frac{1}{2}\Lambda_j(t)^T\mathbf{Y}_j\mathbf{X}_j\mathbf{U}_j^{-1}\mathbf{X}_j^T\mathbf{Y}_j\Lambda_j(t) + (\mathbf{1}+\mathbf{Y}_j\mathbf{X}_j\mathbf{U}^{-1}\xi_j(t))^T\Lambda_j(t) \tag{2.5}$$

$$\mathbf{w}_j(t+1) = \mathbf{U}_j^{-1}(\mathbf{X}_j^T\mathbf{Y}_j\Lambda_j(t+1) - \xi_j(t)) \tag{2.6}$$

$$\zeta_j(t+1) = \zeta_j(t) + \frac{\rho}{2}\sum_{i\in\mathcal{N}_j^*}(\mathbf{w}_j(t+1) - \mathbf{w}_i(t+1)). \tag{2.7}$$

Since each node eliminates some values from its neighborhood in each iteration, the network needs some redundancy to guarantee success. One way of describing the redundancy of the network topology is the following.

Definition 2. *Let J denote a subset of $\mathcal{J}$ and D_J^μ denote the set $\mathcal{D}_J^\mu = \{j \in J : |\mathcal{N}_j \setminus J| \geq \mu\}$. Graph $\mathcal{G} = (\mathcal{J}, \mathcal{E})$ of $M \geq 2$ nodes is μ,ν-robust for $r \geq 0$ and $1 \leq \nu \leq M$ if for every pair of nonempty, disjoint subsets $\bar{J}_1$, $\bar{J}_2$ of J, at least one of the following holds: (i) $|\mathcal{D}_{\bar{J}_1}^\mu| = |\bar{J}_1|$; (ii) $|\mathcal{D}_{\bar{J}_2}^\mu| = |\bar{J}_2|$; and $|\mathcal{D}_{\bar{J}_1}^\mu| + |\mathcal{D}_{\bar{J}_2}^\mu| \geq \nu$.*

The basic idea of this topology assumption is that if the graph is separated into two sets, both sets need to have at least some (ν) nodes that have some (μ) neighbors in the other set. It is obvious that the condition becomes more restrictive as μ or ν increases. For the algorithm to work, it requires that the graph of the network is at least $(b+1, b+1)$-robust. This constraint can be interpreted in a simpler way: no matter how the nodes are divided into two sets, after the screening with respect to b, some nodes from one set will still be able to communicate with some nodes in the other set.

2.2 Experimental results

Next, we show the effectiveness of RD-SVM with numerical experiments. Since SVM is a popular tool for binary classification, the experiment is done on a well-understood binary classification problem, i.e., distinguishing between hand-written digits using SVM. The MNIST dataset [24] is chosen to perform the experiment. First, we generate a graph with 25 nodes, where each pair of nodes has a 0.5 probability to be connected. Then

Table 2.1: Results on MNIST8M dataset

Algorithm	Byzantine nodes	Accuracy	Consensus
DSVM	0	97.76%	YES
DSVM	4	25%	NO
RD-SVM	4	97.75%	YES

4 nodes are randomly picked to be Byzantine nodes. In this experiment, a Byzantine node will broadcast a random vector during each iteration. The training set is made up of digits '0', '1', '6', '7' and every node has access to 200 of each digit. The 4-class classification can be done by 3 rounds of binary classifications: separate '0', '1' and '6', '7'; separate '0' and '1'; separate '6' and '7'. The performance of RD-SVM is compared with the performance of DSVM [21]. Three rounds of experiments are performed: (1) DSVM without failure; (2) DSVM under failure; (3) RD-SVM under failure. The performance is evaluated by the classification accuracy.

The results of the experiments are shown in Table 2.1. It can be shown that DSVM has good performance in the faultless setting but it fails in the presence of Byzantine failures. As apposed to DSVM, RD-SVM maintains good performance when there are Byzantine failures in the network. The comparison indicates the fact that while classic decentralized learning algorithms are not Byzantine resilient, RD-SVM is indeed resilient to Byzantine failures.

2.3 Concluding remarks

RD-SVM is the first algorithm that solves vector-valued decentralized learning problems in the presence of Byzantine failures. We introduce this method to give the reader a demonstration of what Byzantine failure is and how to improve robustness against it. The basic idea is to add a screening stage before each local updating. However, since the screening for RD-SVM is by comparing the hinge loss, it is specifically designed for SVM and cannot be generalized to other learning techniques. Starting from the next chapter, we are going to provide algorithms that can be applied to a general class of learning problems.

Chapter 3

PAC Learnability and the ByRDiE Algorithm

One classic way of solving machine learning problem is to solve the ERM problem (1.2) by using optimization techniques to find the minimizer. It then can be shown that the minimizer of the ERM problem converges to the minimizer of the statistical risk function as the sample size increases. Note that this is valid independent of the optimization techniques. However, it is shown in previous work [25] that the empirical minimization problem (1.2) cannot be exactly solved when there are Byzantine failures in the network, which brings up an obvious but important question: is learning problem solvable under decentralized settings in the presence of Byzantine failures?

In this chapter, we will show that decentralized Byzantine-resilient learning is possible by proposing and analyzing an algorithm named **By**zantine-**R**esilient **D**ecentralized coord**i**nate d**E**scent (ByRDiE). We are going to show that the minimizer learned by ByRDiE is probably and approximately correct (PAC). Instead of solving the ERM problem, we are going to show that ByRDiE statistically solves the learning problem without achieving the empirical minimum.

3.1 PAC learnability over decentralized networks in the presence of Byzantine failures

In order to show that the learning problem is solvable, we first define PAC learnability. If all nonfaulty nodes can find a $\bar{\mathbf{w}}$ such that, for arbitrarily small ϵ and δ, $\mathbf{P}(\|\mathbb{E}[f(\bar{\mathbf{w}},\mathbf{z})]-\mathbb{E}[f(\mathbf{w}^*,\mathbf{z})]\|_2 < \epsilon) \geq 1-\delta$ as a function of N, then $\bar{\mathbf{w}}$ is probably and approximately correct. In words, the model $\mathbf{w}^*$ is PAC learnable if we can learn, at each nonfaulty node, a $\bar{\mathbf{w}}$ that converges to $\mathbf{w}^*$ with high probability.

Before going to the main result, we provide a set of assumptions and definitions that are critical for the analysis in the rest of the dissertation.

Assumption 1. *The risk function $f(\mathbf{w}, \mathbf{z})$ is bounded almost surely over all training samples, i.e.,*

$$f(\mathbf{w}, \mathbf{z}) \leq C < \infty, \forall \mathbf{z} \in \bigcup_{j \in \mathcal{J}} \mathcal{Z}_j. \tag{3.1}$$

Assumption 2. *The risk function $f(\mathbf{w}, \mathbf{z})$ is strictly convex, i.e.,*

$$f(\mathbf{w}_1, \mathbf{z}) > f(\mathbf{w}_2, \mathbf{z}) + \langle \nabla f(\mathbf{w}_2, \mathbf{z}), \mathbf{w}_2 - \mathbf{w}_1 \rangle. \tag{3.2}$$

Assumption 3. *The gradient of $f(\mathbf{w}, \mathbf{z})$ is L-Lipschitz, i.e.,*

$$\|\nabla f(\mathbf{w}_1, \mathbf{z}) - \nabla f(\mathbf{w}_2, \mathbf{z})\| \leq L \|\mathbf{w}_1 - \mathbf{w}_2\|. \tag{3.3}$$

Note that Assumption 3 implies the risk function itself is also Lipschitz, i.e., $\|f(\mathbf{w}_1, \mathbf{z}) - f(\mathbf{w}_2, \mathbf{z})\| \leq L' \|\mathbf{w}_1 - \mathbf{w}_2\|$ for some L [26]. In this dissertation, we focus our analysis on convex functions and give the statistical and algorithmic convergence rate of algorithms on such functions. Although the convergence rate is not given for non-convex functions, later we will numerically show that the algorithms can also work for non-convex functions. Next, we give an assumption on the network topology which is the main difference between distributed setting and decentralized setting.

Definition 3. *A subgraph $\mathcal{G}_r$ of $\mathcal{G}$ is called a reduced graph if it is generated from graph $\mathcal{G}$ by (i) removing all Byzantine nodes along with all their incoming and outgoing edges, and (ii) removing additionally up to b incoming edges from each nonfaulty node. A source component of graph $\mathcal{G}_r$ is a collection of nodes such that each node in the source component has a directed path to every other node in $\mathcal{G}_r$.*

Assumption 4. *All reduced graphs $\mathcal{G}_r$ generated from $\mathcal{G}(\mathcal{J}, \mathcal{E})$ contain a source component of cardinality at least $b + 1$.*

Assumption 4 describes the redundancy of a graph. What it ensures is that after removing a certain number of edges from nonfaulty nodes, each normal node can still receive information from a few other nonfaulty nodes. We emphasize that this topology assumption is sufficient but not necessary in terms of PAC learnability. More specifically, the assumption is required for our Byzantine-resilient algorithms which later will be shown to deliver a PAC learning result. However, other Byzantine-resilient learning techniques may also be able to perform PAC learning with different topology requirements. Nevertheless, consensus-based Byzantine-resilient decentralized algorithms must have some topology requirements to guarantee convergence; see, e.g., [27]. Here is a simple example to explain why. Suppose we have a chain-shaped directed network: node 1 can only receive information from node 2, node 2 can only receive information from node 3, and so on. Without Byzantine nodes, the network can achieve consensus on the local value of node 1 with a trivial algorithm. However, when there is a Byzantine node that is not at either end of the chain, the network will be cut into two groups where nodes from one group cannot receive any information from the other group. In this case, it is theoretically impossible to achieve consensus among all nodes without changing the topology.

Next, we present our main result: the PAC learning problem is solvable for a general class of risk functions.

Theorem 1. *Let Assumptions 1, 2, 3, and 4 be satisfied. There exists an algorithm that can learn a hypothesis* $\bar{\mathbf{w}}$ *at each nonfaulty node that is probably approximately correct in the presence of Byzantine failures. Specifically, with probability at least* $1 - \mathcal{O}(\exp(-N\epsilon^2))$*,*

$$\|\mathbb{E}[f(\bar{\mathbf{w}}, \mathbf{z})] - \mathbb{E}[f(\mathbf{w}^*, \mathbf{z})]\|_2 < \epsilon. \tag{3.4}$$

To the best of our knowledge, Theorem 1 is the first result that shows the PAC learnability of decentralized learning problems in the presence of Byzantine failures. The proof of the theorem and the discussion of the sample complexity are done by giving and analyzing a Byzantine-resilient algorithm called ByRDiE. In the rest of this chapter, we first describe ByRDiE algorithm in detail. We then theoretically and numerically analyze the algorithm to show both the PAC result and the performance of the algorithm.

3.2 Byzantine-resilient decentralized coordinate descent

In the literature, researchers have made progress in solving the scalar-valued Byzantine-resilient decentralized optimization problems [28]. The existing work improves distributed gradient descent (DGD) by adding a screening process before the local update on each node. As a trade-off for robustness, the scalar-valued optimization problem can only be solved inexactly. Our ByRDiE algorithm extends the scalar-valued method to vector-valued cases by adopting coordinate descent to break the vector-cased problem into a series of scalar-valued problems. However, the inexactness in the solution for scalar-valued problems makes it highly non-trivial for both designing and analyzing the algorithm.

3.2.1 Algorithmic details

ByRDiE involves splitting the vector-valued problem into d one-dimensional subproblems using coordinate descent and then solving each scalar-valued subproblem using the Byzantine-resilient approach described in [28]. The exact implementation is detailed in Algorithm 1. The algorithm can be broken into an outer loop (Step 2) and an inner loop (Step 4). The outer loop is the coordinate descent loop, which breaks the vector-valued optimization problem in each iteration r into d scalar-valued subproblems. The inner loop solves a scalar-valued optimization problem in each iteration t and ensures resilience to Byzantine failures. We assume the total number of iterations $\bar{r}$ for coordinate descent are specified during initialization. We use $[\mathbf{w}_j^r(t)]_k$ to denote the k-th element of $\mathbf{w}_j$ at the r-th iteration of the coordinate descent loop and the t-th iteration of the k-th subproblem (coordinate). Without loss of generality, we initialize $[\mathbf{w}_j^1(1)]_k = 0, \forall k = 1, \dots, d$.

We now fix some r and k, and focus on the implementation of the inner loop (Step 4). Every node has some $[\mathbf{w}_j^r(1)]_k$ at the start of the inner loop ($t = 1$). During each iteration t of this loop, all (nonfaulty) nodes engage in the following: *broadcast*, *screening*, and *update*. In the broadcast step (Step 6), all nodes $i \in \mathcal{J}$ broadcast $[\mathbf{w}_i^r(t)]_k$'s and each node $j \in \mathcal{J}$ receives $[\mathbf{w}_i^r(t)]_k, \forall i \in \mathcal{N}_j$. During this step, a node can receive values

Algorithm 1 Byzantine-resilient decentralized coordinate descent

```
Input: $\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_M, \{\rho(\tau)\}_{\tau=1}^{\infty}, b \in \mathbb{N}, \bar{r} \in \mathbb{N}, T \in \mathbb{N}$
 1: Initialize: $r \leftarrow 1$, $t \leftarrow 1$, and $\forall j \in \mathcal{J}', \mathbf{w}_j^r(t) \leftarrow 0$
 2: for $r = 1, 2, \dots, \bar{r}$ do
 3:     for $k = 1, 2, \dots, d$ do
 4:         for $t = 1, 2, \dots, T$ do
 5:             for $j = 1, 2, \dots, |\mathcal{J}'|$ do (in parallel)
 6:                 Receive $[\mathbf{w}_i^r(t)]_k$ from all $i \in \mathcal{N}_j$
 7:                 Find $\mathcal{N}_j^s(r,k,t)$, $\mathcal{N}_j^l(r,k,t)$, $\mathcal{N}_j^*(r,k,t)$ according to (3.5), (3.6), and (3.7)
 8:                 Update $[\mathbf{w}_j^r(t+1)]_k$ as in (3.8)
 9:             end for
10:         end for
11:     end for
12:     $\mathbf{w}_j^{r,T} \leftarrow \mathbf{w}_j^r(T+1), \forall j \in \mathcal{J}'$
13:     $\mathbf{w}_j^{r+1}(1) \leftarrow \mathbf{w}_j^{r,T}, \forall j \in \mathcal{J}'$
14: end for
Output: $\left\{\mathbf{w}_j^{\bar{r},T}\right\}_{j \in \mathcal{J}'}$
```

from both nonfaulty *and* Byzantine neighbors. The main idea of the screening step (Step 7) is to reject values at node j that are either "too large" or "too small" so that the values being used for update by node j in each iteration will be upper and lower bounded by a set of values generated by nonfaulty nodes. To this end, we partition $\mathcal{N}_j$ into 3 subsets $\mathcal{N}_j^*(r,k,t)$, $\mathcal{N}_j^s(r,k,t)$ and $\mathcal{N}_j^l(r,k,t)$, which are defined as following:

$$\mathcal{N}_j^s(r,k,t) = \underset{X: X \subset \mathcal{N}_j, |X|=b}{\arg\min} \sum_{i \in X} [\mathbf{w}_i^r(t)]_k, \tag{3.5}$$

$$\mathcal{N}_j^l(r,k,t) = \underset{X: X \subset \mathcal{N}_j, |X|=b}{\arg\max} \sum_{i \in X} [\mathbf{w}_i^r(t)]_k, \tag{3.6}$$

$$\mathcal{N}_j^*(r,k,t) = \mathcal{N}_j \setminus \mathcal{N}_j^s(r,k,t) \setminus \mathcal{N}_j^l(r,k,t). \tag{3.7}$$

The step is called screening because node j only uses $[\mathbf{w}_i^r(t)]_k$'s from $\mathcal{N}_j^*(r,k,t)$ to update its local variable. Note that there might still be $[\mathbf{w}_i^r(t)]_k$'s received from Byzantine nodes in $\mathcal{N}_j^*(r,k,t)$. We will see later, however, that this does not effect the workings of the overall algorithm.

The final step of the inner loop in ByRDiE is the update step (Step 8). Using $[\nabla \widehat{f}_j(\mathbf{w}_j^r(t))]_k$ to denote the k-th element of $\nabla \widehat{f}_j(\mathbf{w}_j^r(t))$, we can write this update step as follows:

$$[\mathbf{w}_j^r(t+1)]_k = \frac{1}{|\mathcal{N}_j| - 2b + 1} \sum_{i \in \mathcal{N}_j^*(r,k,t) \cup \{j\}} [\mathbf{w}_i^r(t)]_k \\ - \rho(r+t-1)[\nabla \widehat{f}_j(\mathbf{w}_j^r(t))]_k, \tag{3.8}$$

where $\{\rho(\tau)\}_{\tau=1}^{\infty}$ are square-summable (but not summable), diminishing stepsizes: $0 < \rho(\tau+1) \leq \rho(\tau)$, $\sum_\tau \rho(\tau) = \infty$, and $\sum_\tau \rho^2(\tau) < \infty$. Notice that $[\mathbf{w}_j^r(T+1)]_k$ is updated after the k-th subproblem of coordinate descent in iteration r finishes and it stays fixed until the start of the k-th subproblem in the $(r+1)$-th iteration of coordinate descent. An iteration r of the coordinate descent loop is considered complete once all d subproblems within the loop are solved. The local variable at each node j at the end of this iteration is then denoted by $\mathbf{w}_j^{r,T}$ (Step 12). We also express the output of the whole algorithm as $\{\mathbf{w}_j^{\bar{r},T}\}_{j \in J'}$. Finally, note that while Algorithm 1 cycles through the d coordinates of the optimization variables in each iteration r in the natural order, one can use any permutation of $\{1, \ldots, d\}$ in place of this order.

We conclude this discussion by noting that the parameter T in ByRDiE, which can take any value between 1 and ∞, trades off consensus among the nonfaulty nodes and the convergence rate as a function of the number of *communication iterations* $t_c(r, k, t)$, defined as

$$t_c := (r-1)Td + [(k-1)T + t]. \tag{3.9}$$

In particular, given a fixed T, each iteration r of ByRDiE involves Td (scalar-valued) communication exchanges among the neighboring nodes. As will be shown next, we will provide theoretical guarantees for the two extreme cases of $T \to \infty$ and $T = 1$. In the limit of large $\bar{r}$, our results can establish that both extremes result in consensus and convergence to the statistical risk minimizer. In practice, however, different choices of T result in different behaviors as a function of $t_c(\equiv t_c(r, k, t))$, as discussed in the following and as illustrated in our numerical experiments.

When T is large, the two time-scale nature of ByRDiE ensures the disagreement between nonfaulty nodes does not become too large in the initial stages of the algorithm; in particular, the larger the number of iterations T in the inner loop, the smaller the disagreement among the nonfaulty nodes at the beginning of the algorithm. Nonetheless, this comes at the expense of slower convergence to the desired minimizer as a function of the number of communication iterations t_c.

On the other hand, while choosing $T = 1$ also guarantees consensus among nonfaulty nodes, it only does so asymptotically (cf. Theorem 2). Stated differently, ByRDiE cannot guarantee in this case that the disagreement between nonfaulty nodes will be small in the initial stages of the algorithm. This tradeoff between small consensus error and slower convergence (as a function of communication iterations t_c) should be considered by a practitioner when deciding the value of T. The different nature of the two extreme cases requires markedly different proof techniques, which should be of independent interest to researchers. Our discussion so far has focused on the use of a static parameter T within ByRDiE. Nonetheless, it is plausible that one could achieve somewhat better tradeoffs between consensus and convergence through the use of an adaptive parameter T_r in lieu of T that starts with a large value and gradually decreases as r increases. Careful analysis and investigation of such an adaptive two-time scale variant of ByRDiE, however, is beyond the scope of this dissertation.

3.2.2 Theoretical guarantees: Consensus

Our theoretical analysis of ByRDiE focuses on the two extreme cases of $T \to \infty$ and $T = 1$. In both cases, we establish in the following that the output of ByRDiE at nonfaulty nodes converges in probability to the minimum of the statistical risk. Convergence guarantees for a finite-valued $T > 1$ can be obtained from straightforward modifications of the analytical techniques used in the following.

We now turn our attention to theoretical guarantees for ByRDiE. We first show that it leads to consensus among nonfaulty nodes in the network, i.e., all nonfaulty nodes agree on the same variable, in the limit of large $\bar{r}$ and/or T. Then we show in the next sections that the output of ByRDiE converges to the statistical optimum in the limit of large $\bar{r}$ for the two extreme choices of T: $T \to \infty$ and $T = 1$.

Let us begin by establishing the claim of consensus. To this end, focusing exclusively on dimension k in ByRDiE, we see that the k-th dimension of each local variable is updated $\bar{r}T$ times. Fixing any k, we can use an index $m := (r-1)T + t$ to denote the sequence generated for the k-th dimension up to iteration (r,t) in ByRDiE. We define a vector $\Omega(m) \in \mathbb{R}^{|J'|}$ such that $[\mathbf{\Omega}(m)]_j := [\mathbf{w}_j^r(t)]_k$ and let $\boldsymbol{\omega}_j(m)$ denote $[\Omega(m)]_j$. Similarly, we define a vector $\mathbf{G}(m) \in \mathbb{R}^{|J'|}$ such that $[\mathbf{G}(m)]_j := [\nabla \widehat{f}_j(\mathbf{w}_j^r(t))]_k$. Next, let $\bar{\rho}(m) := \rho(r+t-1)$ and note that $\bar{\rho}(m)$ also satisfies $\bar{\rho}(m) \to 0$, $\sum_{m=1}^{\infty} \bar{\rho}(m) = \infty$ and $\sum_{m=1}^{\infty} \bar{\rho}^2(m) < \infty$. We can now express the update of the sequence corresponding to the k-th dimension at nonfaulty nodes in a matrix form as follows:

$$\mathbf{\Omega}(m+1) = \mathbf{Y}(m)\mathbf{\Omega}(m) - \bar{\rho}(m)\mathbf{G}(m), \tag{3.10}$$

where $\mathbf{Y}(m)$ is a matrix that is fully specified in the following.

Let $\mathcal{N}_j'$ and $\mathcal{N}_j^b$ denote the nonfaulty nodes and the Byzantine nodes, respectively, in the neighborhood of $j \in \mathcal{J}'$, i.e., $\mathcal{N}_j' = \mathcal{J}' \cap \mathcal{N}_j$ and $\mathcal{N}_j^b = \mathcal{N}_j \setminus \mathcal{N}_j'$. Notice that one of two cases can happen during each iteration at node $j \in \mathcal{J}'$:

$$\mathcal{N}_j^*(m) \cap \mathcal{N}_j^b \neq \emptyset, \quad \text{or} \tag{3.11(a)}$$

$$\mathcal{N}_j^*(m) \cap \mathcal{N}_j^b = \emptyset. \tag{3.11(b)}$$

For case (3.11(a)), since $|\mathcal{N}_j^b| \leq b$ and $|\mathcal{N}_j^s(m)| = |\mathcal{N}_j^l(m)| = b$, we must have $\mathcal{N}_j^s(m) \cap \mathcal{N}_j' \neq \emptyset$ and $\mathcal{N}_j^l(m) \cap \mathcal{N}_j' \neq \emptyset$. Then for each $i \in \mathcal{N}_j^*(m) \cap \mathcal{N}_j^b$, $\exists s_j^i \in \mathcal{N}_j^s(m) \cap \mathcal{N}_j'$ and $l_j^i \in \mathcal{N}_j^l(m) \cap \mathcal{N}_j'$ satisfying $\boldsymbol{\omega}_{s_j^i}(m) \leq \boldsymbol{\omega}_i(m) \leq \boldsymbol{\omega}_{l_j^i}(m)$. Therefore, we have for each $i \in \mathcal{N}_j^*(m) \cap \mathcal{N}_j^b$ that $\exists \theta_i^j(m) \in [0,1]$ such that the following holds:

$$\boldsymbol{\omega}_i(m) = \theta_i^j(m)\boldsymbol{\omega}_{s_j^i}(m) + (1-\theta_i^j(m))\boldsymbol{\omega}_{l_j^i}(m). \tag{3.12}$$

We can now rewrite the update at node $j \in J'$ as follows:

$$\begin{aligned}\boldsymbol{\omega}_j(m+1) = \frac{1}{|\mathcal{N}_j| - 2b + 1}\Big(\boldsymbol{\omega}_j(m) + \sum_{i \in \mathcal{N}' \cap \mathcal{N}_j^*(m)} \boldsymbol{\omega}_i(m) \\ + \sum_{i \in \mathcal{N}^b \cap \mathcal{N}_j^*(m)} \big(\theta_i^j(m)\boldsymbol{\omega}_{s_j^i}(m) + (1-\theta_i^j(m))\boldsymbol{\omega}_{l_j^i}(m)]_k\big)\Big) + \bar{\rho}(m)[\mathbf{G}(m)]_j.\end{aligned} \tag{3.13}$$

It can be seen from (3.13) that the screening rule in ByRDiE effectively enables nonfaulty nodes to replace data received from Byzantine nodes with convex combinations of data received from nonfaulty nodes in their neighborhoods. This enables us to express the updates at nonfaulty nodes in the form (3.10), with the entries of $Y(m)$ given by

$$[\mathbf{Y}(m)]_{ji} = \begin{cases} \frac{1}{|\mathcal{N}_j|-2b+1}, & i = j, \\ \frac{1}{|\mathcal{N}_j|-2b+1}, & i \in \mathcal{N}' \cap \mathcal{N}_j^*(m), \\ \sum\limits_{n\in\mathcal{N}^b\cap\mathcal{N}_j^*(m)} \frac{\theta_n^j(m)}{|\mathcal{N}_j|-2b+1}, & i \in \mathcal{N}' \cap \mathcal{N}_j^s(m), \\ \sum\limits_{n\in\mathcal{N}^b\cap\mathcal{N}_j^*(m)} \frac{1-\theta_n^j(m)}{|\mathcal{N}_j|-2b+1}, & i \in \mathcal{N}' \cap \mathcal{N}_j^l(m), \\ 0, & \text{otherwise}. \end{cases} \tag{3.14}$$

Notice further that, since $\mathcal{N}_j^*(m) \cap \mathcal{N}_j^b = \emptyset$ (cf. (3.13)), case (3.11(b)) corresponds to a special case of case (3.11(a)) in which we keep only the first, second, and last rows of (3.14). It is worth noting here that our forthcoming proof does not require knowledge of $\mathbf{Y}(m)$; in particular, since the choices of s_j^i and l_j^i are generally not unique, $\mathbf{Y}(m)$ itself is also generally not unique. The main thing that matters here is that $\mathbf{Y}(m)$ will always be a row stochastic matrix; we refer the reader to [29] for further properties of $\mathbf{Y}(m)$.

In order to complete our claim that nonfaulty nodes achieve consensus under ByRDiE, even in the presence of Byzantine failures in the network, fix an arbitrary $m_0 \geq 0$ and consider $m > m_0$. It then follows from (3.10) that

$$\begin{aligned} \mathbf{\Omega}(m+1) &= \mathbf{Y}(m)\mathbf{\Omega}(m) - \bar{\rho}(m)\mathbf{G}(m) \\ &= \mathbf{Y}(m)\mathbf{Y}(m-1)\cdots\mathbf{Y}(m_0)\mathbf{\Omega}(m_0) - \bar{\rho}(m)\mathbf{G}(m) \\ &\quad - \sum_{\tau=m_0}^{m-1} \mathbf{Y}(m)\mathbf{Y}(m-1)\cdots\mathbf{Y}(\tau+1)\bar{\rho}(\tau)\mathbf{G}(\tau). \end{aligned} \tag{3.15}$$

We now define matrices

$$\Phi(m, m_0) := \mathbf{Y}(m)\mathbf{Y}(m-1)\cdots\mathbf{Y}(m_0),$$

$\mathbf{\Phi}(m,m) := \mathbf{Y}(m)$, and $\mathbf{\Phi}(m,m+1) := \mathbf{I}$. Notice that $\mathbf{\Phi}(m,m_0)$ is also row stochastic since it is a product of row-stochastic matrices. We can then express (3.15) as

$$\mathbf{\Omega}(m+1) = \mathbf{\Phi}(m,m_0)\mathbf{\Omega}(m_0) - \sum_{\tau=m_0}^{m} \mathbf{\Phi}(m,\tau+1)\bar{\rho}(\tau)\mathbf{G}(\tau). \tag{3.16}$$

Next, we need two key properties of $\mathbf{\Phi}(m,m_0)$ from [25].

Lemma 1 ([25])**.** *Suppose Assumption 4 holds. Then for any $m_0 \geq 0$, there exists a stochastic vector $\boldsymbol{\pi}(m_0)$ such that*

$$\lim_{m\to\infty} \boldsymbol{\Phi}(m, m_0) = \mathbf{1}\boldsymbol{\pi}^T(m_0). \tag{3.17}$$

In words, Lemma 1 states that the product of the row-stochastic matrices $\mathbf{Y}(m)$ converges to a steady-state matrix whose rows are identical and stochastic. We can also characterize the rate of this convergence; to this end, let ψ denote the total number of reduced graphs that can be generated from $\mathcal{G}$, and define $\nu := \psi|J'|$, $\mathcal{N}_{max} := \max_{j\in\mathcal{J}'} |\mathcal{N}_j|$, and

$$\mu := 1 - \frac{1}{(2\mathcal{N}_{max} - 2b + 1)^{\nu}}.$$

Lemma 2 ([25])**.** *Suppose Assumption 4 holds. We then have that $\forall m_0 \geq 0$,*

$$\Big|[\boldsymbol{\Phi}(m, m_0)]_{ji} - [\boldsymbol{\pi}(m_0)]_i\Big| \leq \mu^{(\frac{m-m_0+1}{\nu})}. \tag{3.18}$$

Lemma 2 describes the rate at which the rows of $\boldsymbol{\Phi}(m, m_0)$ converge to $\boldsymbol{\pi}(m_0)$. We now leverage this result and show that the nonfaulty nodes achieve consensus under ByRDiE in the limit of large m, which translates into $\bar{r} \to \infty$ and/or $T \to \infty$. To this end, under the assumption of $m_0 = 1$, we have from (3.16) the following expression:

$$\boldsymbol{\Omega}(m+1) = \boldsymbol{\Phi}(m, 1)\boldsymbol{\Omega}(1) - \sum_{\tau=1}^{m} \boldsymbol{\Phi}(m, \tau+1)\bar{\rho}(\tau)\mathbf{G}(\tau). \tag{3.19}$$

Next, suppose the nonfaulty nodes stop computing local gradients at time step m and use $\mathbf{G}(m+m') = 0$ for $m' \geq 0$. Then, defining $\mathbf{V}(m) := \lim_{m'\to\infty} \boldsymbol{\Omega}(m+m'+1)$, we obtain:

$$\begin{aligned}\mathbf{V}(m) &= \lim_{m'\to\infty} \boldsymbol{\Phi}(m+m', 1)\boldsymbol{\Omega}(1) - \lim_{m'\to\infty} \sum_{\tau=1}^{m+m'} \boldsymbol{\Phi}(m+m', \tau)\bar{\rho}(\tau)\mathbf{G}(\tau) \\ &= \mathbf{1}\pi^T(1)\boldsymbol{\Omega}(1) - \sum_{\tau=1}^{m-1} \mathbf{1}\boldsymbol{\pi}^T(\tau)\bar{\rho}(\tau)\mathbf{G}(\tau). \end{aligned}\tag{3.20}$$

Notice from (3.20) that all elements in the vector $\mathbf{V}(m) \in \mathbb{R}^{|J'|}$ are identical. Recall that $\mathbf{V}(m)$ is obtained by looking at only one dimension k of the optimization variable in ByRDiE; in the following, we use $v^k(m)$ to denote the identical elements of $\mathbf{V}(m)$ corresponding to dimension k. We then have the following result concerning nonfaulty nodes.

Theorem 2 (Consensus Behavior of ByRDiE). *Let Assumptions 3 and 4 hold and fix $\bar{m} = \bar{r}T + 1$. Then,*

$$\forall j \in J', \forall k \in \{1, \ldots, d\}, \ \left[\mathbf{w}_j^{\bar{r},T}\right]_k \to v^k(\bar{m}) \tag{3.21}$$

as $\bar{r} \to \infty$ and/or $T \to \infty$.

First, let us prove the theorem. Fix any dimension $k \in \{1, \ldots, P\}$ and recall from (3.20) that

$$\begin{aligned}\mathbf{V}(m+1) &= \mathbf{1}\pi^T(1)\Omega(1) \\ &- \sum_{\tau=1}^{m-1} \mathbf{1}\pi^T(\tau+1)\bar{\rho}(\tau)\mathbf{G}(\tau) - \mathbf{1}\pi^T(m+1)\bar{\rho}(m)\mathbf{G}(m).\end{aligned} \tag{3.22}$$

Since $m := (r-1)T + t$ and $[\Omega(m)]_j := [\mathbf{w}_j^r(t)]_k$, we get

$$\begin{aligned}&\mathbf{v}^k(m+1) \\ &= \sum_{i=1}^{|J'|}[\pi(1)]_i[\mathbf{w}_i^1(1)]_k - \sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\sum_{i=1}^{|J'|}[\pi(\tau+1)]_i[\mathbf{G}(\tau)]_i \\ &\qquad - \bar{\rho}(m)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\mathbf{G}(m)]_i.\end{aligned} \tag{3.23}$$

We also get from (3.19) that

$$\begin{aligned}[\mathbf{w}_j^r(t+1)]_k &= \sum_{i=1}^{|J'|}[\Phi(m,1)]_{ji}[\mathbf{w}_i^1(1)]_k \\ &- \sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\sum_{i=1}^{|J'|}[\Phi(m,\tau+1)]_{ji}[\mathbf{G}(\tau)]_i - \bar{\rho}(m)[\mathbf{G}(m)]_j,\end{aligned} \tag{3.24}$$

where we have used the fact that $\Phi(m, m+1) = I$. Thus,

$$\begin{aligned}&\left|[\mathbf{w}_j^r(t+1)]_k - \mathbf{v}^k(m+1)\right| = \\ &\Big|\sum_{i=1}^{|J'|}([\Phi(m,1)]_{ji} - [\pi(1)]_i)[\mathbf{w}_i^1(1)]_k \\ &\quad - \sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\sum_{i=1}^{|J'|}([\Phi(m,\tau+1)]_{ji} - [\pi(\tau+1)]_i)[\mathbf{G}(\tau)]_i \\ &\qquad - \bar{\rho}(m)\sum_{i=1}^{|J'|}[\pi(m+1)]_i([\mathbf{G}(m)]_j - [\mathbf{G}(m)]_i)\Big|,\end{aligned} \tag{3.25}$$

where the last summation follows from the observation that $\sum_{i=1}^{|J'|}[\pi(m+1)]_i = 1$. Further, Assumption 3 implies there exists a coordinate-wise bound $L_\nabla \leq L'$ such that $\forall i, m, |[\mathbf{G}(m)]_i| \leq L_\nabla$. Using this and Lemma 2, we obtain

$$\begin{aligned}
&\Big|[\mathbf{w}_j^r(t+1)]_k - \mathbf{v}^k(m+1)\Big| \\
&\leq \sum_{i=1}^{|J'|}\Big|[\Phi(m,1)]_{ji} - [\pi(1)]_i\Big|\,\big|[\mathbf{w}_i^1(1)]_k\big| \\
&\quad + \sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\sum_{i=1}^{|J'|}\Big|[\Phi(m,\tau+1)]_{ji} - [\pi(\tau+1)]_i\Big|\Big|[\mathbf{G}(\tau)]_i\Big| \\
&\quad + \bar{\rho}(m)\sum_{i=1}^{|J'|}[\pi(m+1)]_i\Big|[\mathbf{G}(m)]_j - [\mathbf{G}(m)]_i\Big| \\
&\leq |J'|L_\nabla\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} + 2\bar{\rho}(m)L_\nabla.
\end{aligned} \tag{3.26}$$

Note that the last inequality in the above expression exploits the fact that $[\mathbf{w}_i^1(1)]_k \equiv 0$. In the case of an arbitrary initialization of ByRDiE, however, we could have bounded the first term in the second inequality in (3.26) as $|J'|\Gamma\mu^{\frac{m+1}{\nu}}$, which still goes to zero as $m \to \infty$.

We now expand the term $\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}}$ in (3.26) as[1]

$$\begin{aligned}
\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} &= \left(\sum_{\tau=1}^{\frac{m}{2}-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} + \sum_{\tau=\frac{m}{2}}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}}\right) \\
&\leq \bar{\rho}(1)\sum_{\tau=1}^{\frac{m}{2}-1}\mu^{\frac{m-\tau}{\nu}} + \bar{\rho}(\tfrac{r}{2})\sum_{\tau=\frac{m}{2}}^{m-1}\mu^{\frac{m-\tau}{\nu}} \\
&= \frac{\bar{\rho}(1)\mu^{\frac{\frac{m}{2}+1}{\nu}}(1-\mu^{\frac{m}{2\nu}})}{\mu^{1-\frac{1}{\nu}}} + \frac{\bar{\rho}(\frac{m}{2})\mu^{\frac{1}{\nu}}(1-\mu^{\frac{m}{2\nu}})}{\mu^{1-\frac{1}{\nu}}}.
\end{aligned}$$

It can be seen from the above expression that $\lim_{m\to\infty}\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} \leq 0$. Since $\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} \geq 0$, it follows that $\lim_{m\to\infty}\sum_{\tau=1}^{m-1}\bar{\rho}(\tau)\mu^{\frac{m-\tau}{\nu}} = 0$. This fact in concert with (3.26) and the diminishing nature of $\bar{\rho}(m)$ give

$$\lim_{m\to\infty}\Big|[\mathbf{w}_j^r(t+1)]_k - \mathbf{v}^k(m+1)\Big| = 0. \tag{3.27}$$

[1]We focus here only on the case of an even m for the sake of brevity; the expansion for an odd m follows in a similar fashion.

Next, take $r = \bar{r}$, $t = T$, and note that $\mathbf{w}_j^r(t+1) = \mathbf{w}_j^{\bar{r},T}$ and $\bar{m} = \bar{r}T + 1 = [(r-1)T + T] + 1 = m + 1$ in this case. Further, $\bar{m} \to \infty$ when $\bar{r} \to \infty$ and/or $T \to \infty$. It therefore follows from (3.27) that $[\mathbf{w}_j^{\bar{r},T}]_k \to \mathbf{v}^k(\bar{m})$ as $\bar{r} \to \infty$ and/or $T \to \infty$. Finally, since k in our analysis was arbitrary, the same holds for all $k = 1, 2, \dots, P$. The proof of Theorem 2 is complete.

Theorem 2 establishes consensus at the nonfaulty nodes under ByRDiE when $\bar{r} \to \infty$ and/or $T \to \infty$. We conclude this discussion by also stating the rate at which the iterates of ByRDiE achieve consensus. To this end, we define the consensus vector $\bar{V}(r) \in \mathbb{R}^P$ in iteration r as $[\bar{V}(r)]_k := v^k(r)$. To keep the notation simple, we limit ourselves to $T = 1$ and use w_j^r to denote $w_j^{r,1}$. Nonetheless, a similar result holds for other values of T.

Theorem 3 (Consensus Rate for ByRDiE)**.** *Let Assumptions 3 and 4 hold. Then, fixing $T = 1$, the iterates of ByRDiE satisfy:*

$$\forall j \in J',\ \|\mathbf{w}_j^r - \bar{\mathbf{V}}(r)\| = \mathcal{O}\left(\sqrt{d}\rho(r)\right), \tag{3.28}$$

where $\bar{\mathbf{V}}(r)$ denotes the consensus vector in iteration r.

Theorem 3, which is a straightforward consequence of the proof of Theorem 2 (cf. (3.26)), guarantees a sublinear rate for consensus; indeed, choosing the stepsize evolution to be $\rho(r) = \mathcal{O}(1/r)$ gives us $\|\mathbf{w}_j^r - \bar{\mathbf{V}}(r)\| = \mathcal{O}\left(\sqrt{d}/r\right)$.

3.2.3 Theoretical guarantee: Convergence for $T \to \infty$

We now move to the second part of the guarantee. This involves showing that the output of ByRDiE converges in probability to the minimizer (and minimum) of the statistical risk for two extreme cases: *Case I:* $T \to \infty$ and *Case II:* $T = 1$. We start our discussion with the case of $T \to \infty$, in which case an auxiliary lemma simply follows from [30, Theorem 2] (also, see [29]).

Lemma 3. *Let Assumptions 3 and 4 hold, and let the k-th subproblem of the coordinate descent loop in iteration r of ByRDiE be initialized with some $\{\mathbf{w}_j\}_{j \in J'}$. Then, as*

$T \to \infty$,

$$\forall j \in \mathcal{J}', \left[\mathbf{w}_j^{r,T}\right]_k \to \operatorname*{arg\,min}_{w' \in \mathbb{R}} \sum_{j \in \mathcal{J}'} \alpha_j(r,k) \widehat{f}_j(\mathbf{w}_j|_{[\mathbf{w}_j]_k = w'}) \tag{3.29}$$

for some $\alpha_j(r,k) \geq 0$ such that $\sum_{j \in J'} \alpha_j(r,k) = 1$.

Lemma 3 shows that each subproblem of the coordinate descent loop in ByRDiE under Case I converges to the minimizer of some convex combination of local empirical risk functions of the nonfaulty nodes with respect to each coordinate. In addition, Lemma 3 guarantees that consensus is achieved among the nonfaulty nodes at the end of each coordinate descent loop under Case I. Note that while this fact is already known from Theorems 2 and 3, Lemma 3 helps characterize the consensus point. In summary, when nonfaulty nodes begin a coordinate descent subproblem with identical local estimates and $T \to \infty$, they are guaranteed to begin the next subproblem with identical local estimates.

We now fix (r,k) and use $\tilde{\mathbf{w}}_k^r$ to denote the identical initial local estimates at nonfaulty nodes at the beginning of k-th subproblem of the coordinate descent loop in the r-th iteration of ByRDiE under Case I. Next, we define $h_k^r(w')$ and $H_k^r(w')$ for $w' \in \mathbb{R}$ as

$$h_k^r(w') := \mathbb{E}[f(\tilde{w}_k^r|_{[\tilde{\mathbf{w}}_k^r]_k = w'})], \quad \text{and} \tag{3.30}$$

$$H_k^r(w') := \sum_{j \in J'} \alpha_j(r,k) \widehat{f}_j(\tilde{\mathbf{w}}_k^r|_{[\tilde{\mathbf{w}}_k^r]_k = w'}) \tag{3.31}$$

for some $\alpha_j(r,k) \geq 0$ such that $\sum_{j \in J'} \alpha_j(r,k) = 1$. Note that $h_k^r(\cdot)$ is strictly convex and Lipschitz continuous. Now for fixed r and k, define

$$w^\star := \operatorname*{arg\,min}_{w' \in \mathbb{R}} h_k^r(w'), \quad \text{and} \tag{3.32}$$

$$\widehat{w} := \operatorname*{arg\,min}_{w' \in \mathbb{R}} H_k^r(w'). \tag{3.33}$$

It should be evident to the reader from (3.30) and (3.32) that the univariate stochastic function $h_k^r(w')$ depends on $\tilde{w}_k^r$ and its (scalar-valued) minimizer $w^\star$, which should not be confused with the vector-valued statistical minimizer w^*, is a function of r and k. Similarly, it should be obvious from (3.31) and (3.33) that $H_k^r(w')$ depends on $\tilde{w}_k^r$

and $\{\alpha_j(r,k)\}_{j\in J'}$, while its minimizer $\widehat{w}$ is also a function of r and k. We are dropping these explicit dependencies here for ease of notation.

In words, if one were to solve the statistical risk minimization problem using (centralized) coordinate descent then $w^\star$ will be the k-th component of the output of coordinate descent after update of each coordinate k in every iteration r. In contrast, $\widehat{w}$ is the k-th component of the outputs of ByRDiE after update of each coordinate k in every iteration r (cf. Lemma 3). While there exist works that relate the empirical risk minimizers to the statistical risk minimizers (see, e.g., [31]), such works are not directly applicable here because of the fact that $H_k^r(w')$ in this paper changes from one pair (r,k) of indices to the next. Nonetheless, we can provide the following uniform statistical convergence result for ByRDiE under Case I that relates the empirical minimizers $\{\widehat{w}\}$ to the statistical minimizers $\{w^\star\}$.

Theorem 4 (Statistical Convergence Rate for ByRDiE)**.** *Let d and $|\mathcal{J}'|$ be fixed, $\bar{r}$ be any (arbitrarily large) positive integer, and $\{\alpha_j(r,k)\geq 0, j\in J'\}_{r,k=1}^{\bar{r},d}$ be any arbitrary collection satisfying $\sum_{j\in\mathcal{J}'}\alpha_j(r,k)=1$. Let $|\widehat{w}|\leq\Gamma$, $|w^\star|\leq\Gamma$, and $\{\tilde{w}_k^r\}_{r,k}\subset W$, and define $\bar{a}:=\max_{(r,k)}\sqrt{\sum_{j\in J'}\alpha_j^2(r,k)}$. Then, as long as Assumptions 1 and 3 hold, we have $\forall\epsilon>0$*

$$\sup_{r,k}\,[h_k^r(\widehat{w})-h_k^r(w^\star)]<\epsilon \tag{3.34}$$

with probability exceeding

$$1-2\exp\left(-\frac{4|\mathcal{J}'|N\epsilon^2}{c_1^2|\mathcal{J}'|\bar{a}^2+\epsilon^2}+|\mathcal{J}'|\log\left(\frac{c_2}{\epsilon}\right)+d\log\left(\frac{c_3}{\epsilon}\right)\right), \tag{3.35}$$

where $c_1:=8C$, $c_2:=24C|\mathcal{J}'|$, and $c_3:=24L'\Gamma d$.

We begin our proof by fixing $w'\in\mathbb{R}$ such that $|\mathbf{w}'|\leq\Gamma$ and defining $\alpha_k^r\in\mathbb{R}^{|J'|}$ as $\alpha_k^r:=[\alpha_j(r,k):j\in J']$ and $\tilde{F}_k^r(w')\in\mathbb{R}^{|J'|}$ as $\tilde{F}_k^r(w'):=[\widehat{f}(\tilde{w}_k^r|_{[\tilde{w}_k^r]_k=w'},S_j):j\in J']$. We can then write

$$H_k^r(w')=\alpha_k^{r\,T}\tilde{F}_k^r(w'). \tag{3.36}$$

Next, notice from (3.30), and (3.31) that

$$\forall(r,k),\ \mathbb{E}[H_k^r(w')]=h_k^r(w'). \tag{3.37}$$

We now fix indices (r,k) and note that the random variables $\alpha_j(r,k)\ell(\tilde{w}_k^r|_{[\tilde{w}_k^r]_k=w'},(x_{jn},y_{jn}))$ involved in the definition of the univariate function $H_k^r(w')$ are (i) independent due to the independence of the training samples, and (ii) bounded as $0 \leq \alpha_j\ell(w,(x_{jn},y_{jn})) \leq \alpha_j C$ due to Assumption 1. Therefore, the following holds $\forall \epsilon' > 0$ due to Hoeffding's inequality [32]:

$$\begin{aligned}\mathbb{P}\left(\left|\alpha_k^{r\,T}\tilde{F}_k^r(w') - h_k^r(w')\right| \geq \epsilon'\right) &\equiv \\ \mathbb{P}\left(\left|H_k^r(w') - h_k^r(w')\right| \geq \epsilon'\right) &\leq 2\exp\left(-\frac{2N\epsilon'^2}{C^2\|\alpha_k^r\|^2}\right). \end{aligned} \tag{3.38}$$

Further, since the $|J'|$-dimensional vector α_k^r is an arbitrary element of the standard simplex, defined as

$$\Delta := \{v \in \mathbb{R}^{|J'|} : \sum_{j=1}^{|J'|}[v]_j = 1 \text{ and } \forall j, [v]_j \geq 0\}, \tag{3.39}$$

the probability bound in (3.38) also holds for any $v \in \Delta$, i.e.,

$$\mathbb{P}\left(\left|v^T\tilde{F}_k^r(w') - h_k^r(w')\right| \geq \epsilon'\right) \leq 2\exp\left(-\frac{2N\epsilon'^2}{C^2\|v\|^2}\right). \tag{3.40}$$

We now define the set $\mathcal{S}_\alpha := \{\alpha_k^r\}_{r,k=1}^{\bar{r},P}$. Our next goal is to leverage (3.40) and derive a probability bound similar to (3.38) that *uniformly* holds for *all* $v \in \mathcal{S}_\alpha$. To this end, let

$$\mathbb{C}_\xi := \{c_1,\dots,c_{d_\xi}\} \subset \Delta \quad \text{s.t.} \quad \mathcal{S}_\alpha \subseteq \bigcup_{q=1}^{d_\xi} B(c_q,\xi) \tag{3.41}$$

denote an ξ-covering of $\mathcal{S}_\alpha$ in terms of the ℓ_2 norm and define $\bar{c} := \arg\max_{c\in\mathbb{C}_\xi}\|c\|$. It then follows from (3.40) and the union bound that

$$\begin{aligned}\mathbb{P}&\left(\sup_{c\in\mathbb{C}_\xi}\left|c^T\tilde{F}_k^r(w') - h_k^r(w')\right| \geq \epsilon'\right) \\ &\qquad\qquad \leq 2d_\xi\exp\left(-\frac{2N\epsilon'^2}{C^2\|\bar{c}\|^2}\right). \end{aligned} \tag{3.42}$$

In addition, we have

$$\begin{aligned}\sup_{v\in\mathcal{S}_\alpha}\left|v^T\tilde{F}_k^r(w') - h_k^r(w')\right| &\overset{(a)}{\leq} \sup_{c\in\mathbb{C}_\xi}\left|c^T\tilde{F}_k^r(w') - h_k^r(w')\right| + \\ &\qquad \sup_{v\in\mathcal{S}_\alpha,c\in\mathbb{C}_\xi}\|v-c\|\|\tilde{F}_k^r(w')\|, \end{aligned} \tag{3.43}$$

where (a) is due to triangle and Cauchy–Schwarz inequalities. Trivially, $\sup_{v\in\mathcal{S}_\alpha, c\in\mathbb{C}_\xi}\|v-c\| \leq \xi$ from the definition of $\mathbb{C}_\xi$, while $\|\tilde{F}_k^r(w')\| \leq \sqrt{|J'|}C$ from the definition of $\tilde{F}_k^r(w')$ and Assumption 1. Combining (4.35) and (4.36), we get

$$\mathbb{P}\left(\sup_{v\in\mathcal{S}_\alpha}\left|v^T\tilde{F}_k^r(w') - h_k^r(w')\right| \geq \epsilon' + \sqrt{|J'|}\xi C\right) \\ \leq 2d_\xi \exp\left(-\frac{2N\epsilon'^2}{C^2\|\bar{c}\|^2}\right). \tag{3.44}$$

We now define $\bar{\alpha} := \arg\max_{v\in\mathcal{S}_\alpha}\|v\|$. It can then be shown from the definitions of $\mathbb{C}_\xi$ and $\bar{c}$ that

$$\|\bar{c}\|^2 \leq 2(\|\bar{\alpha}\|^2 + \xi^2). \tag{3.45}$$

Therefore, picking any $\epsilon'' > 0$, and defining $\epsilon' := \epsilon''/2$ and $\xi := \epsilon''/(2C\sqrt{|J'|})$, we have from (4.37) and (4.38) that

$$\mathbb{P}\left(\sup_{v\in\mathcal{S}_\alpha}\left|v^T\tilde{F}_k^r(w') - h_k^r(w')\right| \geq \epsilon''\right) \\ \leq 2d_\xi \exp\left(-\frac{4|J'|N\epsilon''^2}{4C^2|J'|\|\bar{\alpha}\|^2 + \epsilon''^2}\right). \tag{3.46}$$

In order to obtain the desired uniform bound, we next need to remove the dependence on $\tilde{w}_k^r$ and w' in (4.39). To this end, we write $\tilde{F}_k^r(w')$ and $h_k^r(w')$ as $\tilde{F}_k^r(\tilde{w}_k^r, w')$ and $h_k^r(\tilde{w}_k^r, w')$, respectively, to highlight their dependence on $\tilde{w}_k^r$ and w'. Next, we define

$$\mathbb{U}_\zeta := \{u_1, \ldots, u_{m_\zeta}\} \subset W \quad \text{s.t.} \quad W \subseteq \bigcup_{q=1}^{m_\zeta} B(u_q, \zeta) \tag{3.47}$$

to be a ζ-covering of W in terms of the ℓ_2 norm. It then follows from (4.39) that

$$\mathbb{P}\left(\sup_{v\in\mathcal{S}_\alpha, u\in\mathbb{U}_\zeta}\left|v^T\tilde{F}_k^r(u, [u]_k) - h_k^r(u, [u]_k)\right| \geq \epsilon''\right) \\ \leq 2d_\xi m_\zeta \exp\left(-\frac{4|J'|N\epsilon''^2}{4C^2|J'|\|\bar{\alpha}\|^2 + \epsilon''^2}\right). \tag{3.48}$$

Similar to (4.36), and using notation $\mathcal{L} := \{w' \in \mathbb{R} : |w'| \leq \Gamma\}$, we can also write

$$\begin{aligned}
&\sup_{v\in\mathcal{S}_\alpha, w\in W, w'\in\mathcal{L}} \left|v^T \tilde{F}_k^r(w,w') - h_k^r(w,w')\right| \\
&\leq \sup_{u\in\mathbb{U}_\zeta, v\in\mathcal{S}_\alpha} \left|v^T \tilde{F}_k^r(u,[u]_k) - h_k^r(u,[u]_k)\right| + \\
&\quad \sup_{u\in\mathbb{U}_\zeta, v\in\mathcal{S}_\alpha, w\in W, w'\in\mathcal{L}} \Bigg[\left|v^T \tilde{F}_k^r(w,w') - v^T \tilde{F}_k^r(u,[u]_k)\right| + \\
&\qquad\qquad \left|h_k^r(u,[u]_k) - h_k^r(w,w')\right| \Bigg]. \qquad (3.49)
\end{aligned}$$

Further, since $w|_{[w]_k=w'} \in W$ for any $(w,w') \in W \times \mathcal{L}$, we have from Assumption 3 and definition of the set $\mathbb{U}_\zeta$ that

$$\sup_{u,v,w,w'} \left|v^T \tilde{F}_k^r(w,w') - v^T \tilde{F}_k^r(u,[u]_k)\right| \leq L'\zeta, \quad \text{and} \qquad (3.50)$$

$$\sup_{u,v,w,w'} \left|h_k^r(u,[u]_k) - h_k^r(w,w')\right| \leq L'\zeta. \qquad (3.51)$$

We now fix $\epsilon''' > 0$, and define $\epsilon'' := \epsilon'''/2$ and $\zeta := \epsilon'''/4L'$. We then obtain the following from (4.39)–(3.51):

$$\begin{aligned}
&\mathbb{P}\left(\sup_{v\in\mathcal{S}_\alpha, \{\tilde{w}_k^r\}, w'} \left|H_k^r(w') - h_k^r(w')\right| \geq \epsilon'''\right) \\
&\qquad \leq 2d_\xi m_\zeta \exp\left(-\frac{4|J'|N\epsilon'''^2}{16C^2|J'|\|\bar{\alpha}\|^2 + \epsilon'''^2}\right). \qquad (3.52)
\end{aligned}$$

To conclude, let us define the event

$$\mathcal{A}_\epsilon := \left\{\sup_{v\in\mathcal{S}_\alpha, \{\tilde{w}_k^r\}, w'} \left|H_k^r(w') - h_k^r(w')\right| < \frac{\epsilon}{2}\right\} \qquad (3.53)$$

for any $\epsilon > 0$. Conditioned on this event, we have

$$\begin{aligned}
\forall(r,k),\ h_k^r(\widehat{w}) - h_k^r(w^\star) &= \underbrace{h_k^r(\widehat{w}) - H_k^r(\widehat{w})}_{<\epsilon/2} \\
&+ \underbrace{H_k^r(w^\star) - h_k^r(w^\star)}_{<\epsilon/2} + \underbrace{H_k^r(\widehat{w}) - H_k^r(w^\star)}_{\leq 0} < \epsilon. \qquad (3.54)
\end{aligned}$$

Therefore, given any $\epsilon > 0$, we have from (4.45)–(3.54) that

$$\begin{aligned}
&\mathbb{P}\left(\sup_{r,k}\,[h_k^r(\widehat{w}) - h_k^r(w^\star)] \geq \epsilon\right) \\
&\qquad \leq 2d_\xi m_\zeta \exp\left(-\frac{4|J'|N\epsilon^2}{64C^2|J'|\|\bar{\alpha}\|^2 + \epsilon^2}\right), \qquad (3.55)
\end{aligned}$$

where $\xi = \frac{\epsilon}{8C|J'|}$ and $\zeta = \frac{\epsilon}{8L'}$. The proof now follows from (3.55) and the following facts about the covering numbers of the sets $\mathcal{S}_\alpha$ and W: (1) Since $\mathcal{S}_\alpha$ is a subset of Δ, which can be circumscribed by a sphere in $\mathbb{R}^{|J'|-1}$ of radius $\sqrt{\frac{|J'|-1}{|J'|}} < 1$, we can upper bound d_ξ by $\left(\frac{24C|J'|}{\epsilon}\right)^{|J'|}$ [33]; and (2) Since $W \subset \mathbb{R}^P$ can be circumscribed by a sphere in $\mathbb{R}^P$ of radius $\Gamma\sqrt{P}$, we can upper bound m_ζ by $\left(\frac{24L'\Gamma\sqrt{P}}{\epsilon}\right)^P$. The proof of Theorem 4 is complete.

In words, ignoring minor technicalities that are resolved in Theorem 5 in the following, Theorem 4 states that the coordinate-wise outputs $\{\widehat{w}\}$ of ByRDiE for *all* (r,k) under Case I achieve, with high probability, almost the same statistical risk as that obtained using the corresponding coordinate-wise statistical risk minimizers $\{w^\star\}$. We now leverage this result to prove that the iterates of ByRDiE at individual nodes achieve statistical risk that converges to the minimum statistical risk achieved by the statistical risk minimizer (vector) $\mathbf{w}^*$.

Theorem 5 (Convergence Behavior of ByRDiE). *Let Assumptions 3–4 hold. Then,* $\forall j \in \mathcal{J}', \forall \epsilon > 0,$ *and* $T \to \infty$, *we have*

$$\lim_{\bar{r}\to\infty} \Big[\mathbb{E}[f(\mathbf{w}_j^{\bar{r},T}, (x,y))] - \mathbb{E}[f(\mathbf{w}^*, (x,y))]\Big] < \epsilon \tag{3.56}$$

with probability exceeding

$$1 - 2\exp\left(-\frac{4|\mathcal{J}'|N\epsilon^2}{{c_1'}^2|\mathcal{J}'|\bar{a}^2 + \epsilon^2} + |\mathcal{J}'|\log\left(\frac{c_2'}{\epsilon}\right) + d\log\left(\frac{c_3'}{\epsilon}\right)\right), \tag{3.57}$$

where $c_1' := c_1c_4$, $c_2' := c_2c_4$, *and* $c_3' := c_3c_4$ *for* $c_4 := 2dL\Gamma$, *and* $(\bar{a}, c_1, c_2, c_3)$ *are as defined in Theorem 4.*

Let us begin with any coordinate descent iteration r and dimension k, which starts with some $\tilde{\mathbf{w}}_k^r \in \mathbb{R}^P$ (recall that $\tilde{\mathbf{w}}_1^1 \equiv 0$). In order to facilitate the proof, we explicitly write $w^\star \in \mathbb{R}$, defined in (3.32), and $\widehat{w} \in \mathbb{R}$, defined in (3.33), as ${w^\star}_k^r$ and $\widehat{w}_k^r$, respectively, to bring out their dependence on the indices (r,k). We now define $\widehat{w}^r \in \mathbb{R}^P$ as $[\widehat{\mathbf{w}}^r]_k := \widehat{w}_k^r$ and recall from Lemma 3 and the subsequent definitions that $\forall r \in \mathbb{N}, \forall j \in J', \lim_{T\to\infty} \mathbf{w}_j^{r,T} = \widehat{\mathbf{w}}^r$. It therefore suffices to show that, in the limit of large $\bar{r}$, the statistical risk of $\widehat{\mathbf{w}}^{\bar{r}}$ approaches the minimum statistical risk.

We now fix an arbitrary $\epsilon' \in (0,1)$ and note that

$$\sup_{r,k} [h_k^r(\widehat{w}_k^r) - h_k^r({w^\star}_k^r)] < \epsilon' \tag{3.58}$$

with high probability due to Theorem 4. Note that invoking Theorem 4 requires the conditions $|\widehat{w}_k^r| \leq \Gamma$, $|{w^\star}_k^r| \leq \Gamma$, and $\{\tilde{w}_k^r\}_{r,k} \subset W$. We will return to these conditions in the latter part of the proof. Going forward, we condition on the event described by (3.58) and notice that $\forall w' \in \mathbb{R}$, we have

$$\begin{aligned} h_k^r({w^\star}_k^r) \overset{(a)}{\leq} h_k^r(w') \overset{(b)}{\leq} h_k^r(\widehat{w}_k^{r-1})] + \frac{L}{2}|w' - \widehat{w}_k^{r-1}|^2 \\ + [\nabla h_k^r(\widehat{w}_k^{r-1})]_k (w' - \widehat{w}_k^{r-1}), \end{aligned} \tag{3.59}$$

where (a) follows from the definition of ${w^\star}_k^r$ and (b) follows from Assumption 3. Plugging $w' = \widehat{w}_k^{r-1} - \frac{1}{L}[\nabla h_k^r(\widehat{w}_k^{r-1})]_k$ in (3.59), we obtain

$$h_k^r(\widehat{w}_k^{r-1}) - h_k^r({w^\star}_k^r) \geq \frac{1}{2L}[\nabla h_k^r(\widehat{w}_k^{r-1})]_k^2 \tag{3.60}$$

$$\Leftrightarrow \quad h_k^r(\widehat{w}_k^{r-1}) - h_k^r(\widehat{w}_k^r) > \frac{1}{2L}[\nabla h_k^r(\widehat{w}_k^{r-1})]_k^2 - \epsilon' \tag{3.61}$$

where (3.60) follows from (3.58). We have from (3.61) that $h_k^r(\widehat{w}_k^r)$ is a strict monotonically decreasing function of r for all k as long as $[\nabla h_k^r(\widehat{w}_k^{r-1})]_k^2 \geq 2L\epsilon'$. It therefore follows that there exists some $r_0 \in \mathbb{N}$ such that

$$\forall k, \forall r \geq r_0,\ [\nabla h_k^r(\widehat{w}_k^r)]_k^2 < 4L\epsilon' \tag{3.62}$$

$$\Rightarrow \forall r \geq r_0,\ \|\nabla \mathbb{E}[f(\widehat{\mathbf{w}}^r, (x,y))]\| < 2\sqrt{P}L\epsilon'. \tag{3.63}$$

In addition, convexity of $\mathbb{E}[f(\cdot,(x,y))]$ dictates

$$\begin{aligned} \mathbb{E}[f(\mathbf{w}^*, (x,y))] \geq \mathbb{E}[f(\widehat{\mathbf{w}}^r, (x,y))] \\ + \nabla \mathbb{E}[f(\widehat{\mathbf{w}}^r, (x,y)]^T(\mathbf{w}^* - \widehat{\mathbf{w}}^r). \end{aligned} \tag{3.64}$$

Using (3.63), (3.64), and the Cauchy–Schwarz inequality yields

$$\begin{aligned} \forall r \geq r_0,\ \mathbb{E}[f(\widehat{\mathbf{w}}^r, (x,y))] - \mathbb{E}[f(\mathbf{w}^*, (x,y))] \\ \leq \|\nabla \mathbb{E}[f(\widehat{\mathbf{w}}^r, (x,y)]\| \|\widehat{\mathbf{w}}^r - w^*\| < 2PL\Gamma\epsilon'. \end{aligned} \tag{3.65}$$

Setting $\epsilon' = \epsilon/c_4$ in (3.64) and removing conditioning on (3.58) using Theorem 4 give us the desired bound.

We conclude by commenting on the validity of $|\widehat{w}_k^r| \leq \Gamma$, $|w^{\star}{}_k^r| \leq \Gamma$, and $\{\tilde{\mathbf{w}}_k^r\}_{r,k} \subset W$ needed for Theorem 4. The condition holds for $\tilde{\mathbf{w}}_1^1$ and $w^{\star}{}_1^1$ from the definition of the set W. The proof of Theorem 4 and a union bound argument also tells us that we can augment the bound in Theorem 4 with $\left[h_1^1(\widehat{w}_1^1) - h_1^1(w^{\star}{}_1^1)\right] < \epsilon'$ without either requiring $|\widehat{w}_1^1| \leq \Gamma$ or exploding the probability of failure. This however leads to the condition $h_1^1(\widehat{w}_1^1) < h_1^1(0)$ due to (3.61), which implies $|\widehat{w}_1^1| \leq \Gamma$, $\tilde{\mathbf{w}}_2^1 \in W$, and $|w^{\star}{}_2^1| \leq \Gamma$. We can therefore revert to the original probability bound of Theorem 4 and start over the same argument with the knowledge that $\tilde{\mathbf{w}}_2^1 \in W$, $|\widehat{w}_1^1| \leq \Gamma$, and $|w^{\star}{}_2^1| \leq \Gamma$. The rest of the claim follows by induction.

We now make a couple of remarks concerning Theorem 5. First, note that the uniqueness of the minimum of strictly convex functions coupled with the statement of Theorem 5 guarantee that $\forall j \in \mathcal{J}', \mathbf{w}_j^{\bar{r},T} \to \mathbf{w}^*$ with high probability.

Second, Theorem 5 helps crystallize the advantages of decentralized learning over *local learning*, in which nodes individually solve the empirical risk minimization problem using only their local data samples. Prior work on stochastic convex optimization (see, e.g., [31, Theorem 5 and (11)]) tells us that, with high probability and ignoring the log terms, the gap between the statistical risk achieved by the empirical risk minimizer and the statistical risk minimizer scales as $\mathcal{O}\left(1/\sqrt{\#\text{ of samples}}\right)$ in the centralized setting. This *learning rate* translates into $\mathcal{O}(1/\sqrt{N})$ for local learning and $\mathcal{O}(1/\sqrt{MN})$ for the idealized centralized learning. In contrast, Theorem 5 can be interpreted as resulting in the following *decentralized* learning rate (with high probability):[2]

$$\mathbb{E}[f(\mathbf{w}_j^{\bar{r},T})] - \mathbb{E}[f(\mathbf{w}^*) = \mathcal{O}\left(1/\sqrt{N_{\text{eff}}}\right), \tag{3.66}$$

where $N_{\text{eff}} := N/\bar{a}^2$ denotes the *effective* number of training samples available during decentralized learning.

[2]We are once again ignoring the log terms in our discussion; it can be checked, however, that the log terms resulting from Theorem 5 match the ones in prior works such as [31] on centralized learning.

In order to understand the significance of (3.66), notice that

$$\frac{1}{M} \leq \frac{1}{|\mathcal{J}'|} \leq \bar{a}^2 \leq 1 \;\Rightarrow\; N \leq N_{\text{eff}} \leq |\mathcal{J}'|N \leq NM. \tag{3.67}$$

In particular, $\bar{a}^2 = 1/M$ if and only if there are no Byzantine failures in the network, resulting in the coordinate descent-based decentralized learning of $\mathcal{O}\left(1/\sqrt{NM}\right)$, which matches the centralized learning rate. (This, to the best of our knowledge, is the first result on the explicit learning rate of coordinate descent-based decentralized learning.) In the presence of Byzantine nodes, however, the *maximum* number of *trustworthy* samples in the network is $|\mathcal{J}'|N$, and (3.66) and (3.67) tell us that the learning rate of ByRDiE in this scenario will be somewhere between the idealized learning rate of $\mathcal{O}\left(1/\sqrt{|\mathcal{J}'|N}\right)$ and the local learning rate of $\mathcal{O}\left(1/\sqrt{N}\right)$.

Note that Theorem 5 is of identical form of Theorem 1. By proving Theorem 5, we have also completed the proof of Theorem 1, which means that, under our previous assumptions, a general class of the decentralized learning problems are PAC solvable in the presence of Byzantine failures. Although the ByRDiE algorithm may not be the optimal method, it does apply to all the functions to this class so that, for the first time in the literature, the PAC learnability is shown under Byzantine decentralized settings.

Our discussion so far has focused on the rate of statistical convergence (i.e., learning rate) of ByRDiE. The proof of Theorem 5, however, also contains within itself the algorithmic rate of convergence for ByRDiE. We state this convergence rate in terms of the following theorem, which uses the notation $\bar{f}^0$ to denote the starting statistical risk $\mathbb{E}[f(0)]$, $\bar{f}^*$ to denote the minimum statistical risk $\mathbb{E}[f(\mathbf{w}^*)]$ and $1 - \delta(\epsilon, N, \bar{a})$ to express the probability expression in (3.57).

Theorem 6 (Algorithmic Convergence Rate for ByRDiE)**.** *Let Assumptions 3–4 hold. Then,* $\forall j \in \mathcal{J}', \forall r \in \mathbb{N}, \forall \epsilon > 0,$ *and* $T \to \infty$*, we get with probability exceeding* $1 - \delta(\epsilon, N, \bar{a})$ *that*

$$\mathbb{E}\left[f(\mathbf{w}_j^{r,T})\right] - \bar{f}^* < \max\left\{\left(\bar{f}^0 - \bar{f}^*\right)\left(1 - \frac{r\epsilon}{c_5}\right), \epsilon\right\}, \tag{3.68}$$

where the parameter $\delta(\epsilon, N, \bar{a})$ *is given by*

$$2\exp\left(-\frac{4|\mathcal{J}'|N\epsilon^2}{{c_1'}^2|\mathcal{J}'|\bar{a}^2 + \epsilon^2} + |\mathcal{J}'|\log\left(\frac{c_2'}{\epsilon}\right) + d\log\left(\frac{c_3'}{\epsilon}\right)\right), \tag{3.69}$$

while $c_5 := c_4 L' \sqrt{d}\gamma^*$, and the constants c_1', c_2', c_3', and c_4 are as defined in Theorem 5.

We begin by defining the notation $\bar{f}(\mathbf{w}) := \mathbb{E}[f(\mathbf{w},(x,y))]$ for $\mathbf{w} \in \mathbb{R}^P$. Next, we borrow the notation of $\widehat{\mathbf{w}}^r \in \mathbb{R}^P$ and some facts from the proof of Theorem 5. These facts include the following. First, $\widehat{\mathbf{w}}^r$ is the output of ByRDiE after each iteration r at all nonfaulty nodes, i.e., $\forall r \in \mathbb{N}, \forall j \in J', \lim_{T\to\infty} \mathbf{w}_j^{r,T} = \widehat{\mathbf{w}}^r$. Second, defining $\epsilon' := \epsilon/c_4$, we have

$$\forall (r,k),\ h_k^r(\widehat{w}_k^r) - h_k^r(\widehat{w}_k^{r-1}) < -\epsilon' \tag{3.70}$$

with probability $\geq 1 - \delta(\epsilon, N, \bar{a})$ as long as $\bar{f}(\widehat{\mathbf{w}}^r) - \bar{f}^* > \epsilon$ (see, e.g., (3.61) and the discussion around it). Conditioning on the probability event described by (3.70), the definition of $h_k^r(\cdot)$ and (3.70) then give us the following recursion in r:

$$\forall r,\ \bar{f}(\widehat{\mathbf{w}}^r) - \bar{f}(\widehat{\mathbf{w}}^{r-1}) < -\epsilon' \tag{3.71}$$

as long as $\bar{f}(\widehat{\mathbf{w}}^r) - \bar{f}^* > \epsilon$. Note here that $\widehat{\mathbf{w}}^0 = \tilde{\mathbf{w}}_1^1 \equiv 0$. Therefore, (3.71) and the telescoping sum argument give us

$$\bar{f}(\widehat{\mathbf{w}}^r) < \bar{f}^0 - r\epsilon' \ \Leftrightarrow\ \bar{f}(\widehat{\mathbf{w}}^r) - \bar{f}^* < \bar{f}^0 - \bar{f}^* - r\epsilon' \tag{3.72}$$

$$\Leftrightarrow \bar{f}(\widehat{\mathbf{w}}^r) - \bar{f}^* < \left(\bar{f}^0 - \bar{f}^*\right)\left(1 - \frac{r\epsilon'}{\bar{f}^0 - \bar{f}^*}\right). \tag{3.73}$$

Plugging the inequality $\bar{f}^0 - \bar{f}^* \leq L'\|w^*\| \leq L'\sqrt{P}\gamma^*$ in (3.72) completes the proof.

It can be seen from Theorem 6 that, with high probability, ByRDiE requires $r = \mathcal{O}(1/\epsilon)$ iterations to bring the *excess risk* $\mathbb{E}[f(\mathbf{w}_j^{r,T})] - \bar{f}^*$ down to ϵ. In terms of minimization of the statistical risk, therefore, ByRDiE achieves a sublinear rate of algorithmic convergence with high probability, even in the presence of Byzantine failures in the network.

3.2.4 Theoretical guarantees: Convergence for $T = 1$

Case I for ByRDiE, in which $T \to \infty$, is akin to doing exact line search during minimization of each coordinate, which is one of the classic ways of implementing coordinate descent. Another well-adopted way of performing coordinate descent is to take only one step in the direction of descent in a dimension and then switch to another dimension [34]. Within the context of ByRDiE, this is equivalent to setting $T = 1$ (Case II); our goal here is to provide convergence guarantees for ByRDiE in this case. Our analysis in this section uses the compact notation $\mathbf{w}_j^r := \mathbf{w}_j^{r,1} \equiv \mathbf{w}_j^{r+1}(1)$. In the interest of space, and since the probabilistic analysis of this section is similar to that of the previous section, our probability results are stated asymptotically here, rather than in terms of precise bounds.

The starting point of our discussion is Theorem 2. Recall from Sec. 3.2.2 the definition of the index $m := (r-1)T + t$. When $T = 1$, we have $r = m$ and therefore $[\mathbf{w}_j^r]_k \to v^k(r)$ as $r \to \infty$ according to Theorem 2. In order to provide convergence guarantee, we only need to show that $v^k(r) \xrightarrow{r,N} [\mathbf{w}^*]_k$ in probability. To this end, we define a sequence $\mathbf{Q}(q) \in \mathbb{R}^d$ as follows: $\forall k, [\mathbf{Q}(1)]_k := v^k(1)$, while for the parameterized index $q = (r-1)d + k \neq 1$, $\mathbf{Q}(q)$ is obtained by replacing $[\mathbf{Q}(q-1)]_k$ with $v^k(r)$ and keeping the other dimensions fixed. Similarly, we define a sequence $\eta(q)$ satisfying $\eta(q) = \rho(r)$ for $dr \leq q < (d+1)r$. Notice that $0 < \eta(q+1) \leq \eta(q)$, $\sum_q \eta(q) = \infty$ and $\sum_q \eta^2(q) < \infty$. Since we have from (3.20) that $v^k(r+1) = v^k(r) - \rho(r)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla \widehat{f}(w_i^r, S_i)]_k$, we can write the following iteration:

$$\mathbf{Q}(q+1) = \mathbf{Q}(q) - \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla \widehat{f}_i(\mathbf{w}_i^r)]_k \mathbf{e}_k, \tag{3.74}$$

where $\mathbf{e}_k$ denotes the standard basis vector (i.e., it is zero in every dimension except k and $[\mathbf{e}_k]_k = 1$) and the relationship between $r, k,$ and q is as defined earlier. The sequence $\mathbf{Q}(q)$ effectively helps capture update of the optimization variable after each coordinate-wise update of ByRDiE. In particular, we have the following result concerning the sequence $\mathbf{Q}(q)$.

Lemma 4. *Let Assumptions 1, 3, and 4 hold for ByRDiE and choose $T = 1$. We then have that $\mathbf{Q}(q) \xrightarrow{q,N} \mathbf{w}^*$ in probability.*

Similar to the proof of Theorem 6, we once again use the notation $\bar{f}(\mathbf{Q}(q)) := \mathbb{E}[f(\mathbf{Q}(q),(x,y))]$ to denote the statistical risk incurred by $\mathbf{Q}(q)$ and show $\bar{f}(\mathbf{Q}(q)) \xrightarrow{q,N} \bar{f}(w^*)$ in probability. It then follows from [35, Theorem 4.4] and our assumptions that $\mathbf{w}^*$ is a *strong minimizer* of $\bar{f}(\cdot)$ and, therefore, $\mathbf{Q}(q) \xrightarrow{q,N} \mathbf{w}^*$ in probability.

In order to prove the aforementioned claim, we fix any $\epsilon > 0$ and show that $\bar{f}(\mathbf{Q}(q)) - \bar{f}(w^*) \leq \epsilon$ for all large enough q with probability that approaches 1 as $N \to \infty$. To this end, we claim that $\bar{f}(\mathbf{Q}(q))$ for all q greater than some $q_0 \in \mathbb{N}$ is a strictly monotonically decreasing function with probability 1, which is also lower bounded by $\bar{f}(\mathbf{w}^*)$. By the monotone convergence theorem, therefore, $\bar{f}(\mathbf{Q}(q))$ converges. We claim this convergence only takes place when $|[\nabla \bar{f}(\mathbf{Q}(q))]_k| \leq \epsilon_\nabla$. Relegating the validity of this claim to the latter part of this proof, this means that $|[\nabla \bar{f}(\mathbf{Q}(q))]_k|$ eventually becomes smaller than ϵ_∇ with probability 1 for large enough q, which implies

$$\begin{aligned}
\bar{f}(\mathbf{Q}(q)) - \bar{f}(\mathbf{w}^*) &\leq -\nabla \bar{f}(\mathbf{Q}(q))^T(\mathbf{w}^* - \mathbf{Q}(q)) \\
&\leq \|\nabla \bar{f}(\mathbf{Q}(q))\|_2(\|w^*\|_2 + \|\mathbf{Q}(q)\|_2) \\
&\leq (\sqrt{P}\epsilon_\nabla) \cdot (2\sqrt{P}\Gamma) < \epsilon \qquad (3.75)
\end{aligned}$$

because of convexity of $\bar{f}(\cdot)$, the Cauchy–Schwarz inequality, and our assumptions. Since this is the desired result, we need now focus on the claim of strict monotonicity of $\bar{f}(\mathbf{Q}(q))$ for this lemma. To prove this claim, note from Assumption 3 that

$$\begin{aligned}
\bar{f}(\mathbf{Q}(q+1)) &\leq \bar{f}(\mathbf{Q}(q)) + \nabla \bar{f}(\mathbf{Q}(q))^T(\mathbf{Q}(q+1) - \mathbf{Q}(q)) \\
&\quad + \frac{L}{2}\|\mathbf{Q}(q+1) - \mathbf{Q}(q)\|_2^2 \\
&\overset{(a)}{=} \bar{f}(\mathbf{Q}(q)) + [\nabla \bar{f}(\mathbf{Q}(q))]_k[(\mathbf{Q}(q+1) - \mathbf{Q}(q))]_k \\
&\quad + \frac{L}{2}\big|[\mathbf{Q}(q+1) - \mathbf{Q}(q)]_k\big|^2, \qquad (3.76)
\end{aligned}$$

where (a) follows from the fact that $\mathbf{Q}(q+1) - \mathbf{Q}(q)$ is only nonzero in dimension k.

Next, we rewrite (3.74) as follows:

$$\begin{aligned}
\mathbf{Q}(q+1) &= \mathbf{Q}(q) - \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i\Big([\nabla \widehat{f}(\mathbf{Q}(q), S_i)]_k - \\
&\qquad [\nabla \widehat{f}(\mathbf{Q}(q), S_i)]_k + [\nabla \widehat{f}(\mathbf{w}_i^r, S_i)]_k\Big)\mathbf{e}_k \\
&= \mathbf{Q}(q) - \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla \widehat{f}(\mathbf{Q}(q), S_i)]_k\mathbf{e}_k + \mathbf{E}(q), \qquad (3.77)
\end{aligned}$$

where $\mathbf{E}(q) := \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i\big([\nabla \widehat{f}(\mathbf{Q}(q), S_i)]_k - [\nabla \widehat{f}(\mathbf{w}_i^r, S_i)]_k\big)\mathbf{e}_k$. Plugging this into (4.55) results in

$$\begin{aligned}\bar{f}(\mathbf{Q}(q)) - \bar{f}(\mathbf{Q}(q+1)) \geq & -[\mathbf{E}(q)]_k[\nabla\bar{f}(\mathbf{Q}(q))]_k \\ & + \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k[\nabla\bar{f}(\mathbf{Q}(q))]_k \\ & - \frac{L}{2}\Big|[\mathbf{E}(q)]_k - \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k\Big|^2. \end{aligned} \tag{3.78}$$

The right-hand side of (4.56) strictly lower bounded by 0 implies strict monotonicity of $\bar{f}(\mathbf{Q}(q))$. Simple algebraic manipulations show that this is equivalent to the condition

$$\begin{aligned} & \frac{L\eta(q)^2}{2}\Big(\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k\Big)^2 \\ & < \eta(q)\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k[\nabla\bar{f}(\mathbf{Q}(q))]_k \\ & + L\eta(q)[\mathbf{E}(q)]_k\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k \\ & - [\mathbf{E}(q)]_k[\nabla\bar{f}(\mathbf{Q}(q))]_k - \frac{L}{2}[\mathbf{E}(q)]_k^2. \end{aligned} \tag{3.79}$$

Next, notice $\mathbb{E}[\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k] = \mathbb{E}[\nabla f(\mathbf{Q}(q),(x,y))]_k$. We now make an assumption whose validity is also discussed at the end of the proof. We assume $\exists q_0' \in \mathbb{N} : \forall q \geq q_0', \mathbf{Q}(q) \in W$, in which case we can show using arguments similar to the ones in the proof of Theorem 4 that

$$\mathbb{P}\big(|\sum_{i\in J'}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k - [\nabla\bar{f}(\mathbf{Q}(q))]_k| \leq \epsilon'\big) \tag{3.80}$$

converges to 1 uniformly for all (r,k) (equivalently, all q) for any $\epsilon' > 0$. We therefore have with probability 1 (as $N \to \infty$)

$$\begin{aligned}\Big(\sum_{i=1}^{|J'|}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k\Big)^2 \leq & [\nabla\bar{f}(\mathbf{Q}(q))]_k^2 + \epsilon'^2 \\ & + 2\epsilon'\Big|[\nabla\bar{f}(\mathbf{Q}(q))]_k\Big|. \end{aligned} \tag{3.81}$$

Next, we consider two cases: (i) $[\nabla\bar{f}(\mathbf{Q}(q))]_k > 0$, and (ii) $[\nabla\bar{f}(\mathbf{Q}(q))]_k < 0$. When $[\nabla\bar{f}(\mathbf{Q}(q))]_k > 0$, we have in probability $\sum_{i\in J'}[\pi(r+1)]_i[\nabla\widehat{f}(\mathbf{Q}(q),S_i)]_k \geq [\nabla\bar{f}(\mathbf{Q}(q))]_k - \epsilon'$. This fact along with (4.58), the realization that $\frac{L\eta(q)^2}{2} - \eta(q) < 0$ for large enough

q since $\eta(q) \to 0$, and some tedious but straightforward algebraic manipulations show that the following condition is sufficient for (4.57) to hold in probability:

$$\begin{aligned}
&\left|[\nabla \bar{f}(\mathbf{Q}(q))]_k\right| > \\
&\left(\frac{2\epsilon'\left|[\nabla \bar{f}(\mathbf{Q}(q))]_k\right|}{2 - L\eta(q)} + \frac{2\left|[\mathbf{E}(q)]_k\right|\left|[\nabla \bar{f}(\mathbf{Q}(q))]_k\right|}{2\eta(q) - L\eta(q)^2}\right. \\
&+ \frac{2L\left|[E(q)]_k\right|\left|\sum_{i \in J'}[\pi(r+1)]_i[\nabla \widehat{f}(\mathbf{Q}(q), S_i)]_k\right|}{2 - L\eta(q)} \\
&\left.+ \frac{L\left|[\mathbf{E}(q)]_k\right|^2}{2\eta(q) - L\eta(q)^2} + \frac{2L\eta(q)\epsilon'\left|[\nabla \bar{f}(\mathbf{Q}(q))]_k\right|}{2 - L\eta(q)} + \frac{L\eta(q)\epsilon'^2}{2 - L\eta(q)}\right)^{\frac{1}{2}}.
\end{aligned} \tag{3.82}$$

Using similar arguments, one can also show that the case $[\nabla \bar{f}(\mathbf{Q}(q))]_k < 0$ also results in (3.82) as a sufficient condition for (4.57) to hold in probability. We now note that $|[\nabla \bar{f}(\cdot)]_k|$ and $|[\nabla \widehat{f}(\cdot,\cdot)]_k|$ in (3.82) can be upper bounded by some constants $L_{\bar{\nabla}}$ and L_{∇} by virtue of Assumption 3 and the definition of W. This results in the following sufficient condition for (4.57):

$$\begin{aligned}
\left|[\nabla \bar{f}(\mathbf{Q}(q))]_k\right| > &\left(\frac{2L_{\bar{\nabla}}\epsilon' + 2LL_{\bar{\nabla}}\eta(q)\epsilon' + L\eta(q)\epsilon'^2}{2 - L\eta(q)}\right. \\
&\left.+ \frac{2LL_{\nabla}\left|[\mathbf{E}(q)]_k\right|}{2 - L\eta(q)} + \frac{2L_{\bar{\nabla}}\left|[\mathbf{E}(q)]_k\right| + L\left|[\mathbf{E}(q)]_k\right|^2}{2\eta(q) - L\eta(q)^2}\right)^{\frac{1}{2}}.
\end{aligned} \tag{3.83}$$

The right-hand side of (4.62) can be made arbitrarily small (and, in particular, equal to ϵ_{∇}) through appropriate choice of ϵ' and large enough q; indeed, we have from our assumptions, and the definitions of $\eta(q)$ and $\mathbf{E}(q)$ that both $[\mathbf{E}(q)]_k$ and $[\mathbf{E}(q)]_k/\eta(q)$ converge to 0 as $q \to \infty$.

This completes the proof, except that we need to validate one remaining claim and discuss one assumption. The claim is that $\bar{f}(\mathbf{Q}(q))$ cannot converge when $|[\nabla \bar{f}(\mathbf{Q}(q))]_k| > \epsilon_{\nabla}$. We prove this by contradiction. Suppose $\exists k \in \{1, \dots, P\}$ and $\epsilon_0 > 0$ such that $|[\nabla \bar{f}(\mathbf{Q}(q))]_k| - \epsilon_{\nabla} > \epsilon_0$ for all q. We know $\exists q_0 \in \mathbb{N}$ such that the right hand side of (4.62) becomes smaller than ϵ_{∇} for all $q \geq q_0$. Therefore, adding ϵ_0 to the right hand side of (4.62) and combining with (4.56) gives $\forall q \geq q_0$:

$$\bar{f}(\mathbf{Q}(q)) - \bar{f}(\mathbf{Q}(q+1)) \geq (2\eta(q) - L\eta(q)^2)(\epsilon_0^2 + 2\epsilon_{\nabla}\epsilon_0). \tag{3.84}$$

Taking summation on both sides of (4.63) from $q = q_0$ to ∞, and noting that $\sum_{q=q_0}^{\infty} \eta(q) =$

∞ and $\sum_{q=q_0}^{\infty} \eta(q)^2 < \infty$, gives us $\bar{f}(\mathbf{Q}(q_0)) - \lim_{q\to\infty} \bar{f}(\mathbf{Q}(q)) = \infty$. This contradicts the fact that $\bar{f}(\cdot)$ is lower bounded, thereby validating our claim.

Finally, the assumption $\exists q_0' \in \mathbb{N} : \forall q \geq q_0', \mathbf{Q}(q) \in W$ is true with probability 1 (as $N \to \infty$) by virtue of the facts that W is defined in terms of the sublevel set of $\bar{f}(\cdot)$, (3.80) holds $\forall q < q_0'$ without requiring the assumption, $\exists q_0 \in \mathbb{N}$ such that $\bar{f}(\mathbf{Q}(q))$ is monotonic in q for all $q \geq q_0$ due to (3.80), and the probabilistic "onion peeling" induction argument at the end of the proof of Theorem 5 is applicable in this case also (except that one will have to start the argument from some index $q = q_0' \geq q_0$).

We are now ready to state the convergence result for ByRDiE under Case II (i.e, $T = 1$).

Theorem 7 (Asymptotic Convergence of ByRDiE)**.** *Let Assumptions 1, 3, and 4 hold for ByRDiE and choose $T = 1$. Then, $\forall j \in J'$, $\mathbf{w}_j^{\bar{r}} \xrightarrow{\bar{r},N} \mathbf{w}^*$ in probability.*

Proof. We have from Theorem 2 that $\forall j \in J', [\mathbf{w}_j^{\bar{r}}]_k \xrightarrow{\bar{r}} v^k(\bar{r})$ for all $k \in \{1, 2, \ldots, d\}$. The definition of $\mathbf{Q}(q)$ along with Lemma 4 also implies that $v^k(\bar{r}) \xrightarrow{\bar{r},N} [\mathbf{w}^*]_k$ in probability. This completes the proof of the theorem. $\square$

3.3 Numerical analysis

In this section, we validate our theoretical results and make various observations about the performance of ByRDiE using two sets of numerical experiments. The first set of experiments involves learning of a binary classification model from the *infinite MNIST* dataset[3] that is distributed across a network of nodes. This set of experiments fully satisfies all the assumptions in the theoretical analysis of ByRDiE. The second set of experiments involves training of a small-scale neural network for classification of the *Iris* dataset [36] distributed across a network. The learning problem in this case corresponds to a nonconvex one, which means this set of experiments does not satisfy the main assumptions of our theorems. Nonetheless, we show in the following that ByRDiE continues to perform well in such decentralized nonconvex learning problems.

[3] `https://leon.bottou.org/projects/infimnist`

3.3.1 Decentralized SVM using Infinite MNIST dataset

We consider a decentralized linear binary classification problem involving MNIST handwritten digits dataset. The (infinite) MNIST dataset comprises images of handwritten digits from '0' to '9'. Since our goal is the demonstration of the usefulness of ByRDiE in the presence of Byzantine failures, we focus only on decentralized training of a linear support vector machine (SVM) for classification between digits '5' and '8', which tend to be the two most inseparable digits. In addition to highlighting the robustness of ByRDiE against Byzantine failures in this problem setting, we evaluate its performance for different choices of the parameters T, N, and b.

In terms of the experimental setup, we generate Erdős–Rényi networks ($p = 0.5$) of M nodes, b of which are randomly chosen to be Byzantine nodes. All nonfaulty nodes are allocated N samples—equally divided between the two classes—from the dataset, while a Byzantine node broadcasts random data uniformly distributed between 0 and 1 to its neighbors in each iteration. When running the ByRDiE algorithm, each node updates one dimension T times before proceeding to the next dimension. All tests are performed on the same test set with 1000 samples of digits '5' and '8' each.

We first report results that confirm the idea that ByRDiE can take advantage of cooperation among different nodes to achieve better performance even when there are Byzantine failures in the network. This involves varying the local sample size N and comparing the classification accuracy on the test data. We generate a network of $M = 50$ nodes, randomly pick $b = 10$ nodes within the network to be Byzantine nodes (20% failures), vary N from 10 to 30, and average the final set of results over 10 independent (over network, Byzantine nodes, and data allocation) Monte Carlo trials of ByRDiE. The performance of ByRDiE is compared with two approaches: (*i*) coordinate descent-based training using only local data (local CD); and (*ii*) decentralized gradient descent-based [20] training involving network data (DGD). To achieve the best convergence rate for ByRDiE, T is chosen to be 1 in these experiments. The final set of results are shown in Fig 3.1, in which the average classification accuracy is plotted against the number of algorithmic iterations, corresponding to the number of (scalar-valued) communication iterations for ByRDiE, the number of per-dimension updates for local CD, and the number of (vector-valued) communication iterations for DGD. It can be seen that the performance of local CD is not good enough due to the small local sample size. On the other hand, when trying to improve performance by cooperating among different nodes, DGD fails for lack of robustness against Byzantine failures. In contrast, the higher accuracy of ByRDiE shows that ByRDiE can take advantage of the larger distributed dataset while being Byzantine resilient.

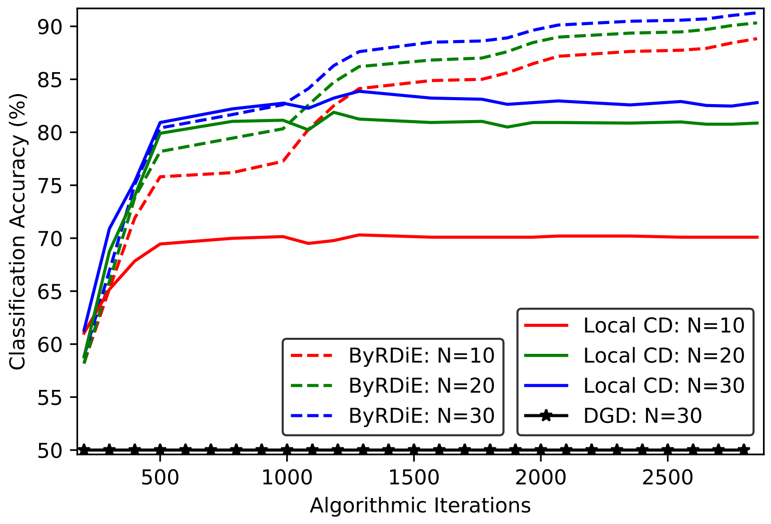
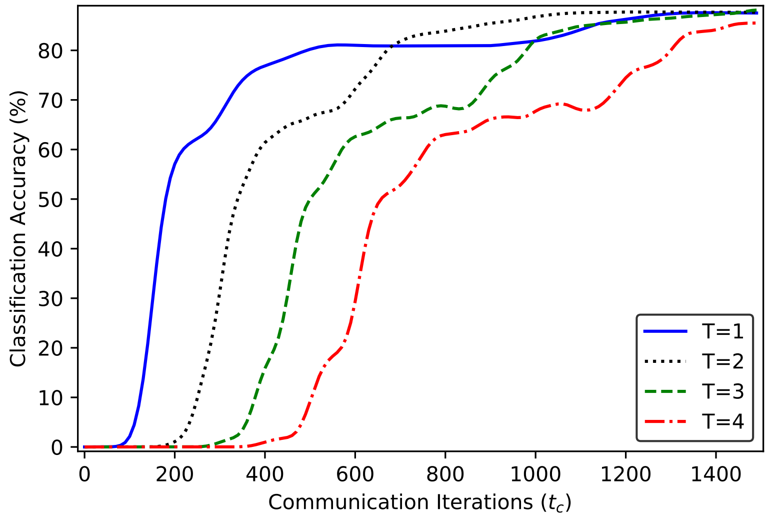
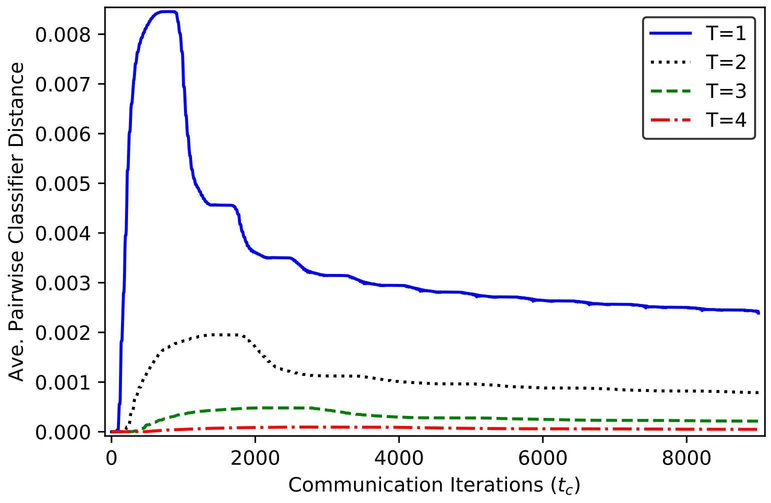


Figure 3.1: Average classification accuracy of ByRDiE, local CD, and DGD (on test data) for different values of N as a function of the number of algorithmic iterations for decentralized training of a linear SVM on (infinite) MNIST dataset.

Next, we investigate the impact of different values of T in ByRDiE on the tradeoff between consensus and convergence rate. This involves generating a network of $M = 50$ nodes that includes randomly placed $b = 5$ Byzantine nodes within the network (10% failures), randomly allocating $N = 60$ training samples to each nonfaulty node, and averaging the final set of results over 10 independent trials. The corresponding results are reported in Fig. 3.2 for four different values of T as a function of the number of communication iterations t_c in terms of (i) average classification accuracy (Fig. 3.2a) and (ii) average pairwise distances between local classifiers (Fig. 3.2b). It can be seen from these figures that $T = 1$ leads to the fastest convergence in the initial stages of the algorithm, but this fast convergence comes at the expense of the largest differences among local classifiers, especially in the beginning of the algorithm. In contrast, while $T = 4$ results in the slowest convergence, it ensures the closeness of the local classifiers at all stages of the algorithm.



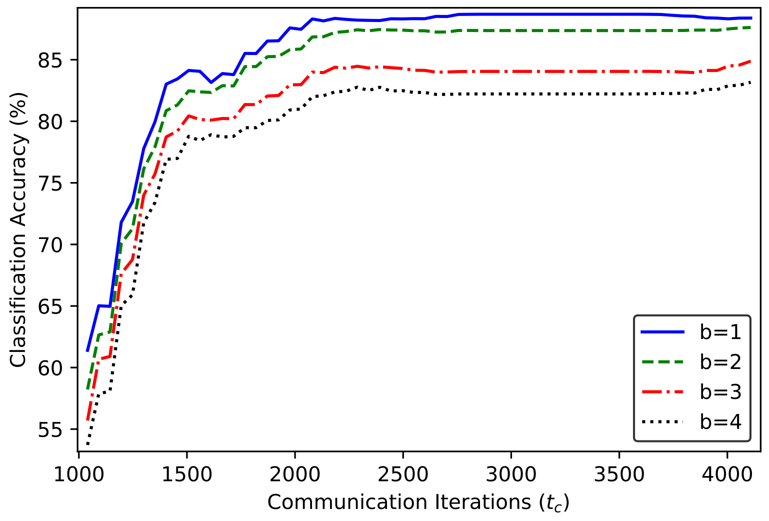
(a) Classification accuracy for different values of T



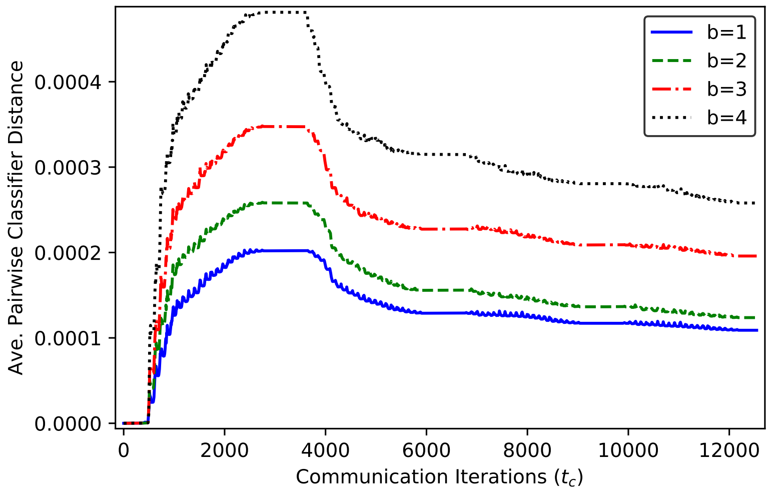
(b) Consensus behavior for different values of T

Figure 3.2: Convergence and consensus behavior of ByRDiE as a function of the number of communication iterations t_c for different values of the parameter T. All plots correspond to decentralized training of a linear SVM using the MNIST dataset.

Finally, we investigate the impact of different values of b (the actual number of Byzantine nodes) on the performance of ByRDiE. In order to amplify this impact, we focus on a smaller network of $M = 20$ nodes with a small number of $N = 10$ samples per node and $T = 3$. Assuming resilience against the worst-case Byzantine scenario, ByRDiE requires the neighborhood of each node to have at least $(2b+1)$ nodes. Under this constraint, we find that $b \geq 5$ in this setting, which translates into 25% or more of the nodes as being Byzantine, often renders ByRDiE unusable. We therefore report our results in terms of both consensus and convergence behavior by varying b from 1 to 4. The final results, averaged over 10 independent trials, are shown in Fig. 3.3. It can be seen from these figures that both the classification accuracy (Fig. 3.3a) and the consensus performance (Fig. 3.3b) of ByRDiE suffer as b increases from 1 to 4. Such behavior, however, is in line with our theoretical developments. First, as b increases, the *post-screening* graph becomes sparser, which slows down information diffusion and consensus. Second, as b increases, fewer samples are incorporated into decentralized learning, which limits the final classification accuracy of decentralized SVM.



(a) Classification accuracy for different values of b



(b) Consensus behavior for different values of b

Figure 3.3: Convergence and consensus behavior of ByRDiE within a small network ($M = 20$) as a function of the number of communication iterations t_c for different number b of Byzantine nodes in the network. All plots correspond to decentralized training of a linear SVM using the MNIST dataset.

3.3.2 Decentralized neural network using Iris dataset

While the theoretical guarantees for ByRDiE have been developed for convex learning problems, we now demonstrate the usefulness of ByRDiE in decentralized nonconvex learning problems. Our setup in this regard involves decentralized training of a small-scale neural network using the three-class Iris dataset. This dataset consists of 50 samples each from three species of irises, with each sample being a four-dimensional feature vector. While one of the species in this data is known to be linearly separable from the other two, the remaining two species cannot be linearly separated. We employ a fully connected three-layer neural network for classification of this dataset, with a four-neuron input layer, a three-neuron hidden layer utilizing the *rectified linear unit* (ReLU) activation function and a three-neuron output layer using the softmax function for classification. The decentralized setup corresponds to a random network of $M = 10$ nodes with one Byzantine node ($b = 1$), in which each nonfaulty node is allocated $N = 15$ samples that are equally divided between the three classes.

Algorithm	**Iterations to achieve 95% accuracy**	**Consensus**
DGD	∞	No
ByRDiE	19	Yes
Centralized CD	17	N/A

Table 3.1: Decentralized training of a neural network in the presence of Byzantine failures using ByRDiE and DGD.

Since decentralized training of this neural network requires solving a decentralized nonconvex problem, our theoretical guarantees for ByRDiE do not hold in this setting. Still, we use ByRDiE with $T = 1$ to train the neural network for a total of 200 independent trials with independent random initializations. Simultaneously, we use DGD for decentralized training and also train the neural network in a centralized setting using CD for comparison purposes. The final results are reported in Table 3.1, which lists the average number of *outer* iterations needed by each training algorithm to achieve 95% classification accuracy. It can be seen from these results that while DGD completely breaks down in the presence of a *single* Byzantine node, ByRDiE continues to be resilient to Byzantine failures in decentralized nonconvex learning problems and comes close to matching the performance of centralized CD in this case.

3.4 Concluding remarks

In this chapter, we have explicitly answered the question: is PAC learning possible in a decentralized fashion when there are Byzantine failures in the network? We have verified the possibility for PAC learning by providing a Byzantine-resilient decentralized learning algorithm, i.e., ByRDiE. Although ByRDiE is the first valid solution for a general class of vector-valued Byzantine-resilient decentralized learning problems, we believe that the algorithmic performance of ByRDiE is not optimal and thus can be improved. In the next chapter, we are going to investigate the possibility to improve the performance of Byzantine-resilient decentralized learning.

Chapter 4

Byzantine-resilient Decentralized Gradient Descent (BRIDGE)

Although ByRDiE can be used to learn a PAC result for a general class of the risk functions, it is not the optimal solution to all these learning problems. The main idea of ByRDiE is to break the vector-valued problem into a series of scalar-valued subproblems by employing coordinate descent. Then the scalar-valued problems can be approximately solved using one-dimensional algorithms similar to [27, 28]. Since the (model) aggregation rules in these works require a scalar sorting process, the algorithms cannot be applied to models in unordered spaces, e.g., vectors and matrices. On the other hand, if the vector-input risk function does not separate into independent scalar-input functions, optimizing the function separately along each coordinate usually does not guarantee the minimum of the vector-valued problem. ByRDiE strictly enforces one-coordinate-a-time updates because it also requires scalar sorting process, which prevents block coordinate descent from being used. Since consensus is required, every coordinate has to be updated often enough, which prevents random coordinate descent being used. Therefore, when the learning model, such as in deep neural networks, lies in a high-dimensional space, it is in general too costly to calculate separate gradients with respect to each dimension at each node, which makes ByRDiE not feasible for large-scale learning problems.

To overcome the drawbacks of one-coordinate-a-time update pattern of ByRDiE, in this chapter, we introduce a gradient-descent-based Byzantine-resilient decentralized learning framework termed **B**yzantine-**r**es**i**lient **d**ecentralized **g**radient d**e**scent (BRIDGE). The main difference between BRIDGE and ByRDiE is that BRIDGE exchanges and updates a whole vector at each iteration as opposed to the multiple scalar exchanges and updates for ByRDiE. Therefore, BRIDGE has less local computation cost at each node (due to gradient computation) and carries fewer communication overheads associated with package transmissions. We are going to show, later in the numerical experiments, that BRIDGE can be applied to learning models such as deep neural networks for which existing algorithms are not feasible. We also provide different variants of BRIDGE based on different aggregation rules and we give algorithmic and statistical convergence rates for one of the variants, which matches the rate of ByRDiE. To the best of our knowledge, BRIDGE is the first Byzantine-resilient decentralized learning algorithm that can be applied to high-dimensional learning models such as deep neural networks.

4.1 Algorithmic details

Algorithm 2 Byzantine-resilient decentralized gradient descent (BRIDGE)

```
Input: $b \in \mathbb{N}$, $\mathcal{Z}_j$, and $\{\rho(t)\}_{t=1}^{\infty}$ at node $j \in \mathcal{J}'$
1: $t \leftarrow 0$, $\mathbf{w}_j(0) \leftarrow \mathbf{0}$
2: for $t = 0, 1, 2, ...$ do
3:     Broadcast $\mathbf{w}_j(t)$
4:     Receive $\mathbf{w}_i(t)$ from $i \in \mathcal{N}_j$
5:     $\mathbf{y}_j(t) \leftarrow \mathsf{screen}(\{\mathbf{w}_i(t)\}_{i \in \mathcal{N}_j \cup \{j\}})$
6:     $\mathbf{w}_j(t+1) \leftarrow \mathbf{y}_j(t) - \rho(t)\nabla f_j(\mathbf{w}_j(t))$
7: end for
Output: $\mathbf{w}_j(t)$
```

When there is no Byzantine failures in the network, one way of solving decentralized learning problems is to let each node update its local variable $\mathbf{w}_j(t)$ as

$$\mathbf{w}_j(t+1) = \sum_{i \in \mathcal{N}_j \cup \{j\}} a_{ji}\mathbf{w}_i(t) - \rho(t)\nabla f_j(\mathbf{w}_j(t)), \tag{4.1}$$

where a_{ji} is the weight and $\{\rho(t)\}$ is a positive sequence satisfying $\rho(t+1) \leq \rho(t)$, $\sum_{t=0}^{\infty} \rho(t) \to \infty$ and $\sum_{t=0}^{\infty} \rho(t)^2 < \infty$. This is known as decentralized gradient descent (DGD) [20]. The main difference between the proposed algorithm and the classic DGD method is that there is a screening step before each update, which is the key to make BRIDGE Byzantine resilient. The process at each node $j \in \mathcal{J}'$ is as shown in Algorithm 2. When initializing the algorithm, it is necessary to specify b, the maximum number of Byzantine nodes that the algorithm should be able to tolerate. Each node $j \in \mathcal{J}'$ initializes at $\mathbf{w}_j(0) = 0$ or some arbitrary vector. During each iteration t, node j first broadcasts $\mathbf{w}_j(t)$ and then receives $\mathbf{w}_i(t)$ from all $i \in \mathcal{N}_j$. Next, node j performs a screening among all $\mathbf{w}_i(t)$'s. Here we introduce several variants of BRIDGE based on different screening methods. The main motivation for these screening rules come from the literature on robust statistics [37] and all these rules have appeared in some form in robust averaging consensus [38, 39] and robust distributed learning [12, 15, 40, 41].

BRIDGE-T uses coordinate-wise trimmed mean as screening. Similar screening idea is employed in distributed frameworks [15] and decentralized frameworks [27, 28]. The idea of trimmed mean is to remove the b largest and the b smallest values in each coordinate from the neighborhood and use the average of the rest for update. Specifically, at each iteration t, for each coordinate k in parallel, BRIDGE-T finds three sets:

$$\overline{\mathcal{N}}_j^k(t) = \underset{X:\{X \in \mathcal{N}_j, |X|=b\}}{\arg\min} \sum_{i \in X} [\mathbf{w}_i(t)]_k, \tag{4.2}$$

$$\underline{\mathcal{N}}_j^k(t) = \underset{X:\{X \in \mathcal{N}_j, |X|=b\}}{\arg\max} \sum_{i \in X} [\mathbf{w}_i(t)]_k, \tag{4.3}$$

and

$$\mathcal{N}_j^k(t) = \mathcal{N}_j \setminus \overline{\mathcal{N}}_j^k(t) \setminus \underline{\mathcal{N}}_j^k(t). \tag{4.4}$$

The k-th element of the screening function output is

$$[\mathbf{y}_j(t)]_k = \frac{1}{|\mathcal{N}_j| - 2b + 1} \sum_{i \in \mathcal{N}_j^k(t) \cup \{j\}} [\mathbf{w}_i(t)]_k. \tag{4.5}$$

BRIDGE-T requires each node having at least $2b + 1$ neighbors. Note that elements from different neighbors may survive the screening at different coordinates. The average is not taken over vectors. Therefore the calculation of $\mathbf{y}(t)$ has to be coordinate-wise separated.

BRIDGE-M uses coordinate-wise median as screening. Similar screening idea is employed in distributed frameworks [15] and decentralized frameworks [38]. Similar to

BRIDGE-T, BRIDGE-M is also a coordinate-wise separated process where

$$[\mathbf{y}_j(t)]_k = \mathsf{median}(\{[\mathbf{w}_i(t)]_k\}_{i\in\mathcal{N}_j\cup\{j\}}). \tag{4.6}$$

The process needs to be applied to vectors in a coordinate-wise manner because median for vectors is not well-defined.

BRIDGE-K uses the Krum function as screening. Similar screening idea is employed in the distributed frameworks [12]. The Krum function is described as the following. We denote $h \sim i$ for $i, h \in \mathcal{N}_j \cup \{j\}$ if $\mathbf{w}_h$ is one of the $|\mathcal{N}_j| - b - 2$ vectors with least $\|\mathbf{w}_h - \mathbf{w}_i\|$. Then we find the neighbor index i^* by

$$i_j^*(t) = \arg\min_{i\in\mathcal{N}_j} \sum_{h\sim i} \|\mathbf{w}_h(t) - \mathbf{w}_i(t)\|. \tag{4.7}$$

The screening output then will be $\mathbf{y}_j(t) = \mathbf{w}_{i^*}(t)$. Unlike BRIDGE-T and BRIDGE-M, BRIDGE-K is a vector-valued operation so that the surviving vector will be entirely from one neighbor. BRIDGE-K requires the neighborhood of all nodes being larger than $2b + 2$.

BRIDGE-B uses a combination of Krum and trimmed mean as screening. Similar screening idea has been employed in distributed frameworks [41]. BRIDGE-B first selects $|\mathcal{N}_j| - 2b$ neighbors by recursively applying Krum function and removing the selected vector from the candidate list. Then coordinate-wise trimmed mean is applied on the selected $|\mathcal{N}_j| - 2b$ neighbors as described in BRIDGE-T. Intuitively, Krum guarantees the surviving neighbors are close to most of the neighbors as whole vectors and trimmed mean guarantees that the output of screening is close to surviving neighbors in each coordinate. The cost of this strict screening is that BRIDGE-B requires each node having more than $4b + 3$ neighbors.

4.2 Theoretical analysis of BRIDGE-T

In this section, we provide the statistical and algorithmic convergence rates of BRIDGE-T. Before going to the main theorem, we provide one more assumption for the class of learning problems.

Assumption 5. *The risk function $f(\mathbf{w}, \mathbf{z})$ is λ-strongly convex, i.e.,*

$$f(\mathbf{w}_1, \mathbf{z}) \geq f(\mathbf{w}_2, \mathbf{z}) + \langle \nabla f(\mathbf{w}_2, \mathbf{z}), \mathbf{w}_2 - \mathbf{w}_1 \rangle + \frac{\lambda}{2}\|\mathbf{w}_1 - \mathbf{w}_2\|^2.$$

In order to show derive the convergence rate of BRIDGE-T, we require strong convexity for the risk function and later we will show that the convergence depends on the convexity parameter λ. Next, we give our main theoretical result on BRIDGE-T.

Theorem 8. *If Assumption 1, 3, 4, and 5 are satisfied, BRIDGE-T can achieve consensus on all nonfaulty nodes, i.e., $\mathbf{w}_j(t) = \mathbf{w}_i(t)$ $\forall i, j \in \mathcal{J}'$, as $t \to \infty$. Further, as $N \to \infty$, the output of BRIDGE-T converges sublinearly in t to the minimum of the global statistical risk at each nonfaulty node. Specifically, given an $\epsilon > 0$, $\forall i, j \in \mathcal{J}'$, with probability exceeding*

$$1 - 2\exp\left(- \frac{4|\mathcal{J}'|N\epsilon^2}{16L^2|\mathcal{J}'|d\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon^2} + |\mathcal{J}'|\log\left(\frac{12L\sqrt{|\mathcal{J}'|d}}{\epsilon}\right) + d\log\left(\frac{12L'\Gamma\sqrt{d}}{\epsilon}\right)\right), \tag{4.8}$$

$$\|\mathbf{w}_j(t+1) - \mathbf{w}^*\| \leq \sqrt{1 - \lambda\rho(t)}\|\mathbf{w}_j(t) - \mathbf{w}^*\| + \mathcal{O}\left(\sqrt{d}\rho(t)\right) + \epsilon, \tag{4.9}$$

where Γ is a finite constant depending on the starting point and $\bar{\boldsymbol{\alpha}}$ is a problem-dependent (unknown) vector whose elements are non-negative and sum up to 1.

The probability bound in (4.8) indicates that with probability at least $1 - \delta$, $\epsilon = \mathcal{O}\left(\sqrt{\frac{\|\bar{\boldsymbol{\alpha}}\|^2 \log \frac{2}{\delta}}{N}}\right)$. When $N \to \infty$ and choosing $\rho(t)$ as an $\mathcal{O}(1/t)$ sequence, (4.9) leads to a sublinear convergence rate. Both the algorithmic and statistical convergence rate match ByRDiE. If there is no failure in the network, the non-resilient algorithms such as DGD usually have statistical learning rate $\mathcal{O}\left(\sqrt{1/MN}\right)$. If each node runs centralized algorithm with the given N samples, the learning rate is $\mathcal{O}\left(\sqrt{1/N}\right)$. Since $\bar{\boldsymbol{\alpha}}$ is a stochastic vector, $1/MN \leq \|\bar{\boldsymbol{\alpha}}\|^2/N \leq 1/N$. The theorem shows that BRIDGE-T reduces the sample complexity by a factor of $\|\bar{\boldsymbol{\alpha}}\|^2$ for each node by cooperating over a network but it cannot beat DGD in the fault-free case. This is the trade-off between sample complexity and robustness.

Among all the variants, BRIDGE-T is the closest to ByRDiE. Here we make some comparisons between the two algorithms. In terms of theoretical convergence rate, both

ByRDiE and BRIDGE-T have a sublinear convergence rate. This is due to the fact that the consensus process is of order $\mathcal{O}(\rho(t))$, which is the main bottleneck in consensus-based decentralized learning algorithms. To achieve consensus, every coordinate has to be updated frequent enough. However, because ByRDiE is coordinate-descent-based, to update all the d coordinates, ByRDiE requires each node calculating local gradients d times. Without any separability assumption, the calculation cannot be done in parallel. When d is large, the local computation overhead is very obvious. For example, in applications such as deep neural networks, d can easily be in millions. One iteration of ByRDiE requires running forward propagation millions of times, which makes it unfeasible. BRIDGE-T, on the other hand, is a gradient-descent-based algorithm, which only requires one round of forward and backward propagation to update all coordinates of the model. Therefore, although BRIDGE-T and ByRDiE have similar convergence rate in t, BRIDGE-T is roughly d times cheaper in terms of local computational costs. Besides computational cost, in decentralized applications, communication efficiency is also an important consideration. To transmit the whole vector, ByRDiE transfers d scalar messages while BRIDGE-T transfers one d-dimensional vector. In terms of quantization, transmitting one d-dimensional vector which with vector quantization will always outperform transmitting d scalars. In terms of communications overhead, transmitting d scalars requires d headers while transmitting one d-dimensional vector requires only one.

4.2.1 Convergence analysis

While gradient descent is well understood in the literature, we observe that BRIDGE does not take a regular gradient step at each iteration. The main idea of proving Theorem 8 is to take advantage of the convergence property of gradient descent and try to bound the distance between one gradient descent step and one BRIDGE step. The proof can be briefly described as the following. The update of $\mathbf{w}_j(t)$ can be described as, for each k in parallel,

$$[\mathbf{w}_j(t+1)]_k = \frac{1}{|\mathcal{N}_j| - 2b + 1} \sum_{i \in \mathcal{N}_j^k(t) \cup \{j\}} [\mathbf{w}_i(t)]_k - \rho(t)[\nabla f_j(\mathbf{w}_j(t))]_k. \tag{4.10}$$

We first define a vector sequence $\mathbf{v}(t)$ and show that $\mathbf{w}_j(t) \to \mathbf{v}(t)$ as $t \to \infty$, which is the proof for consensus. We then consider three sequences $\mathbf{x}(t)$, $\mathbf{u}(t)$, and $\mathbf{v}(t)$ that will be defined later. We define the following distances: $\|\mathbf{x}(t+1) - \mathbf{w}^*\| = a_1$, $\|\mathbf{u}(t+1) - \mathbf{x}(t+1)\| = a_2$, $\|\mathbf{v}(t+1) - \mathbf{u}(t+1)\| = a_3$, and $\|\mathbf{w}_j(t+1) - \mathbf{v}(t+1)\| = a_4$. Observe that $\|\mathbf{w}_j(t+1) - \mathbf{w}^*\| \leq a_1 + a_2 + a_3 + a_4$, we then show that a_1, a_2, a_3 and a_4 all go to 0. This is the proof for optimality.

We first show the consensus result. As $t \to \infty$, for some $\mathbf{v}(t)$,

$$a_4 = \|\mathbf{w}_j(t) - \mathbf{v}(t)\| \leq \sqrt{d}|\mathcal{J}'|C_w\mu^{\frac{t}{\nu}} + \sqrt{d}|\mathcal{J}'|L\sum_{\tau=0}^{t}\rho(\tau)\mu^{\frac{t-\tau+1}{\nu}} \to 0. \tag{4.11}$$

The convergence in (4.11) can also be interpreted as $\mathbf{w}_j(t) \to \mathbf{v}(t)$ as $t \to \infty$. The proof is as the following.

Recall that the update is done in parallel for all coordinates. So we pick one coordinate k and prove that all nodes achieve consensus in this coordinate. Since k is arbitrarily picked, we then conclude that consensus is achieved for all coordinates. In this section, we drop the index k for all variables for simplicity. It should be straight forward that the variables are k-dependent.

Define a vector $\mathbf{\Omega}(t) \in \mathbb{R}^{|\mathcal{J}'|}$ whose elements are the k-th elements of $\mathbf{w}_j(t)$ from nonfaulty nodes only, i.e., $[\mathbf{\Omega}(t)]_j = [\mathbf{w}_j(t)]_k \ \forall j \in \mathcal{J}'$. We first show that the update can be written in a matrix form which only involves nonfaulty nodes, i.e.,

$$\mathbf{\Omega}(t+1) = \mathbf{Y}(t)\mathbf{\Omega}(t) - \rho(t)\mathbf{g}(t), \tag{4.12}$$

where $\mathbf{g}(t)$ is formed as $[\mathbf{g}(t)]_j = [\nabla f_j(\mathbf{w}_j(t))]_k$. The formulation of matrix $\mathbf{Y}(t)$ can be described as following. Let $\mathcal{N}_j'$ denote the nonfaulty nodes in the neighborhood of node j, i.e., $\mathcal{N}_j' = \mathcal{J}' \cap \mathcal{N}_j$. The set of Byzantine neighbors can be defined as $\mathcal{N}_j^b = \mathcal{N}_j \setminus \mathcal{N}_j'$. One of two cases can happen during each iteration, (i) $\mathcal{N}_j^k(t) \cap \mathcal{N}_j^b \neq \emptyset$ or (ii) $\mathcal{N}_j^k(t) \cap \mathcal{N}_j^b = \emptyset$. To make the expression clear, we drop the iteration indicator t for the rest of this discussion. It should be straightforward that the variables are t-dependent. For case (i), since $|\mathcal{N}_j^b| \leq b$ and $|\mathcal{N}_j^{k'}| = b$, we know that $\mathcal{N}_j^{k'} \cap \mathcal{N}_j' \neq \emptyset$. Similarly, $\mathcal{N}_j^{k''} \cap \mathcal{N}_j' \neq \emptyset$. Then $\exists m_j' \in \mathcal{N}_j^{k'} \cap \mathcal{N}_j'$ and $m_j'' \in \mathcal{N}_j^{k''} \cap \mathcal{N}_j'$ satisfying $[\mathbf{w}_{m_j'}]_k < [\mathbf{w}_i]_k < [\mathbf{w}_{m_j''}]_k$ for any $i \in \mathcal{N}_j^k$. So that for each $i \in \mathcal{N}_j^k \cap \mathcal{N}_j^b$, $\exists \theta_i \in (0,1)$

satisfying $[\mathbf{w}_i]_k = \theta_i[\mathbf{w}_{m'_j}]_k + (1-\theta_i)[\mathbf{w}_{m''_j}]_k$. In this way, we can express the update with only messages from nonfaulty nodes. The elements of matrix Y can be written as

$$[\mathbf{Y}]_{ji} = \begin{cases} \frac{1}{|\mathcal{N}_j|-2b+1}, & i \in \mathcal{N}' \cap \mathcal{N}_j^k, \\ \frac{1}{|\mathcal{N}_j|-2b+1}, & i = j, \\ \sum\limits_{i' \in \mathcal{N}^b \cap \mathcal{N}_j^k} \frac{\theta_{i'}}{|\mathcal{N}_j|-2b+1}, & i = m'_j, \\ \sum\limits_{i' \in \mathcal{N}^b \cap \mathcal{N}_j^k} \frac{1-\theta_{i'}}{|\mathcal{N}_j|-2b+1,} & i = m''_j, \\ 0, & \text{else.} \end{cases} \tag{4.13}$$

For case (ii), since all nodes in $\mathcal{N}_j^k$ are already nonfaulty, we keep only the first, second and last rows of (4.13). Note that since the choices of m'_j and m''_j are generally not unique, the formulation of matrix $\mathbf{Y}$ is also not unique. So far, we have expressed the update of nonfaulty nodes in matrix form involving only nonfaulty nodes.

Next, define a transition matrix to represent the product of $\mathbf{Y}(t)$,

$$\mathbf{\Phi}(t, t_0) = \mathbf{Y}(t)\mathbf{Y}(t-1)...\mathbf{Y}(t_0). \tag{4.14}$$

Let ψ be the total number of reduced graphs we can generate from $\mathcal{G}$. Let $\nu = \psi|\mathcal{J}'|$. Denote $\max\limits_{j\in\mathcal{J}'}|\mathcal{N}_j|$ by $\mathcal{N}_{max}$. Let $\mu = 1 - \frac{1}{(2\mathcal{N}_{max}-2b+1)^{\nu}}$. Then it is know from previous work [25, 29] that

$$[\mathbf{\Phi}(t, t_0)]_{ji} - [\boldsymbol{\alpha}(t_0)]_i \le \mu^{(\frac{t-t_0+1}{\nu})}, \tag{4.15}$$

where $\boldsymbol{\alpha}(t_0)$ satisfies $[\boldsymbol{\alpha}(t_0)]_j \ge 0$ and $\sum\limits_{j=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}(t_0)]_j = 1$. It can also be expressed as

$$\lim_{t\to\infty} \mathbf{\Phi}(t, t_0) = \mathbf{1}\boldsymbol{\alpha}^T(t_0). \tag{4.16}$$

Taking $t = 0$ as the starting point, we can express the iterations as

$$\begin{aligned}
\mathbf{\Omega}(1) &= \mathbf{Y}(0)\mathbf{\Omega}(0) - \rho(0)\mathbf{g}(0) \\
\mathbf{\Omega}(2) &= \mathbf{Y}(1)\mathbf{\Omega}(1) - \rho(1)\mathbf{g}(1) \\
&= \mathbf{Y}(1)\mathbf{Y}(0)\mathbf{\Omega}(0) - \mathbf{Y}(1)\rho(0)\mathbf{g}(0) - \rho(1)\mathbf{g}(1) \\
\dots \\
\mathbf{\Omega}(t) &= \mathbf{Y}(t-1)\mathbf{\Omega}(t-1) - \rho(t-1)\mathbf{g}(t-1) \\
&= \mathbf{Y}(t-1)\mathbf{Y}(t-2)\cdots\mathbf{Y}(0)\mathbf{\Omega}(0) - \sum_{\tau=0}^{t-1}\mathbf{Y}(t-1)\mathbf{Y}(t-2)\dots\mathbf{Y}(\tau+1)\rho(\tau)\mathbf{g}(\tau) \\
&= \mathbf{\Phi}(t,0)\mathbf{\Omega}(0) - \sum_{\tau=0}^{t-1}\mathbf{\Phi}(t-1,\tau+1)\rho(\tau)\mathbf{g}(\tau). \qquad (4.17)
\end{aligned}$$

Let us create a scenario that all nodes stop computing local gradients after iteration t so that $\mathbf{g}(\tau) = 0$ when $\tau > t$. Define a vector $\bar{\mathbf{v}}(t)$ under this scenario, i.e.,

$$\begin{aligned}
\bar{\mathbf{v}}(t) &= \lim_{T\to\infty}\mathbf{\Omega}(t+T+1) \\
&= \lim_{T\to\infty}\mathbf{\Phi}(t+T,0)\mathbf{\Omega}(0) - \lim_{T\to\infty}\sum_{\tau=0}^{t+T}\mathbf{\Phi}(t+T,\tau)\rho(\tau)\mathbf{g}(\tau) \\
&= \mathbf{1}\boldsymbol{\alpha}^T(0)\mathbf{\Omega}(0) - \sum_{\tau=0}^{t+T}\mathbf{1}\boldsymbol{\alpha}^T(\tau)\rho(\tau)\mathbf{g}(\tau) \\
&= \mathbf{1}\boldsymbol{\alpha}^T(0)\mathbf{\Omega}(0) - \sum_{\tau=0}^{t-1}\mathbf{1}\boldsymbol{\alpha}^T(\tau)\rho(\tau)\mathbf{g}(\tau). \qquad (4.18)
\end{aligned}$$

Observe that $\bar{\mathbf{v}}$ has identical elements in all dimensions. Let scalar sequence $v(t)$ denote one element of $\bar{\mathbf{v}}$. Next, we show that $[\mathbf{w}_j(t)]_k \to v(t)$ as $t \to \infty$. From (4.18),

$$v(t) = \sum_{i=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}(0)]_i[w_i(0)]_k - \sum_{\tau=0}^{t-1}\rho(\tau)\sum_{i=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}(\tau)]_i[\nabla f_i(\mathbf{w}_i(\tau))]_k \qquad (4.19)$$

Then recall from the update of $[w_j(t)]_k$ that

$$[\mathbf{w}_j(t)]_k = \sum_{i=1}^{|\mathcal{J}'|}[\mathbf{\Phi}(t-1,0)]_{ji}[\mathbf{w}_i(0)]_k - \sum_{\tau=0}^{t-1}\rho(\tau)\sum_{i=1}^{|\mathcal{J}'|}[\mathbf{\Phi}(t-1,\tau)]_{ji}[\nabla f_i(\mathbf{w}_i(\tau))]_k \quad (4.20)$$

If Assumption 3 holds and we initiate the algorithm from some vector with finite norm, we can always find two scalars C_w and L satisfying that $\forall j \in \mathcal{J}'$, $|[\mathbf{w}_j(0)]_k| \le C_w$ and

$|[\nabla f_j(\mathbf{w}_j)]_k| \le L$. Then we have

$$\begin{aligned}
|[\mathbf{w}_j(t)]_k - v(t)| &\le |\sum_{i=1}^{|\mathcal{J}'|}([\mathbf{\Phi}(t-1,0)]_{ji} - [\boldsymbol{\alpha}(0)]_i)[\mathbf{w}_i(0)]_k| + \\
&|\sum_{\tau=0}^{t-1}\rho(\tau)\sum_{i=1}^{|\mathcal{J}'|}([\mathbf{\Phi}_k(t-1,\tau)]_{ji} - [\boldsymbol{\alpha}(\tau)]_i)[\nabla f_i(\mathbf{w}_i(\tau))]_k| \\
&\le |\mathcal{J}'|C_w\mu^{\frac{t}{\nu}} + |\mathcal{J}'|L\sum_{\tau=0}^{t}\rho(\tau)\mu^{\frac{t-\tau+1}{\nu}} \to 0 \qquad (4.21)
\end{aligned}$$

as $t \to 0$. Since k is arbitrarily picked, the convergence is true for all dimensions. Define a vector $\mathbf{v}(t)$ satisfying $[\mathbf{v}(t)]_k = v(t)$ for $1 \le k \le d$. This then implies (4.11).

It follows from (4.11) that the rate of consensus convergence is $\mathcal{O}(\sqrt{d}\rho(t))$. Specifically, if choosing $\rho(t)$ to be $\mathcal{O}(1/t)$ gives us $\|\mathbf{w}_j(t) - \mathbf{v}(t)\| = \mathcal{O}(\sqrt{d}/t)$.

From (4.11), we have an upper bound for d_4. We then bound the other distances to show $\mathbf{w}_j(t) \to w^*$. Note that the sequence $\mathbf{v}(t)$ is not truly kept at any node, so we first describe the "update" of $\mathbf{v}(t)$. The update for the full vector can be written in the form

$$\mathbf{v}(t+1) = \mathbf{v}(t) - \rho(t)\mathbf{g}_1(t) \qquad (4.22)$$

where $\mathbf{g}_1(t)$ satisfies $[\mathbf{g}_1(t)]_k = \sum_{i=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}^k(t)]_i[\nabla f_i(\mathbf{w}_i)]_k$ for $1 \le k \le P$. Define another vector $\mathbf{g}_2(t)$ satisfying $[\mathbf{g}_2(t)]_k = \sum_{i=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}^k(t)]_i[\nabla f_i(\mathbf{v}(t))]_k$. We define a new sequence $\mathbf{u}(t+1)$ as

$$\mathbf{u}(t+1) = \mathbf{v}(t) - \rho(t)\mathbf{g}_2(t). \qquad (4.23)$$

Recalling that

$$a_3 = \|\mathbf{v}(t+1) - \mathbf{u}(t+1)\| = \|\mathbf{g}_2(t) - \mathbf{g}_1(t)\|, \qquad (4.24)$$

from (4.21) and Assumption 3 we have

$$a_3 \le \sqrt{P}|\mathcal{J}'|L'C_w\mu^{\frac{t}{\nu}} + \sqrt{P}|\mathcal{J}'|LL'\sum_{\tau=0}^{t}\rho(\tau)\mu^{\frac{t-\tau+1}{\nu}}. \qquad (4.25)$$

Next, defining a new sequence $\mathbf{X}(t)$ as

$$\mathbf{x}(t+1) = \mathbf{v}(t) - \rho(t)\nabla\mathbb{E}[f(\mathbf{v}(t))], \qquad (4.26)$$

we have

$$a_2 = \|\mathbf{u}(t+1) - \mathbf{x}(t+1)\| = \|\mathbf{g}_2(t) - \mathbb{E}[\nabla f(\mathbf{v}(t))]\|. \tag{4.27}$$

Here we give a lemma to show the relationship between $\mathbf{g}_2(t)$ and the gradient of the statistical risk.

Lemma 5. *If Assumption 1 and 3 are satisfied, with probability at least* $1-\delta$,

$$a_2 \leq \sup_t |\mathbf{g}_2(\mathbf{v}(t)) - \mathbb{E}[\nabla f(\mathbf{v}(t))]| = \mathcal{O}\left(\sqrt{\frac{\|\bar{\boldsymbol{\alpha}}\|^2 \log \frac{2}{\delta}}{N}}\right) \tag{4.28}$$

where $\bar{\boldsymbol{\alpha}} \in \mathbb{R}^{|\mathcal{J}'|}$ *satisfies* $[\boldsymbol{\alpha}]_j \geq 0$ *and* $\sum_{j=1}^{|\mathcal{J}'|} [\boldsymbol{\alpha}]_j = 1$.

To prove the Lemma, we first observe at some dimension k,

$$\mathbb{E}[\mathbf{g}_2(t)]_k = \mathbb{E}\sum_{i=1}^{|\mathcal{J}'|}[\boldsymbol{\alpha}^k(t)]_i[\nabla f_i(\mathbf{v}(t))]_k = \mathbb{E}[\nabla f(\mathbf{v}(t))]_k. \tag{4.29}$$

Since k is arbitrarily picked, it is also true that

$$\mathbb{E}[\mathbf{g}_2(t)] = \mathbb{E}[\nabla f(\mathbf{v}(t))] \tag{4.30}$$

Note that $\mathbf{v}(t)$ depends on t and $\boldsymbol{\alpha}^k(t)$ depends on both t and k. We need to show that the convergence is simultaneously true for all $\mathbf{v}(t)$ and $\boldsymbol{\alpha}^k(t)$. We fix one coordinate k and drop the index k for simplicity. We define a vector $\mathbf{h}(t)$ as $\mathbf{h}(t) := [\nabla f_j(\mathbf{v}(t)) : j \in \mathcal{J}']$. Then $\mathbf{g}_2(t) = \langle \boldsymbol{\alpha}(t), \mathbf{h}(t)\rangle$. We know from Hoeffding's inequality [32]:

$$\mathbb{P}\big(|\langle \boldsymbol{\alpha}(t), \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon\big) \leq 2\exp\left(-\frac{2N\epsilon^2}{L^2\|\boldsymbol{\alpha}(t)\|^2}\right) \tag{4.31}$$

Further, since the $|\mathcal{J}'|$-dimensional vector $\boldsymbol{\alpha}(t)$ is an arbitrary element of the standard simplex, defined as

$$\Delta := \{\mathbf{q} \in \mathbb{R}^{|\mathcal{J}'|} : \sum_{j=1}^{|\mathcal{J}'|}[\mathbf{q}]_j = 1 \text{ and } \forall j, [\mathbf{q}]_j \geq 0\}, \tag{4.32}$$

the probability bound in (4.31) also holds for any $\mathbf{u} \in \Delta$, i.e.,

$$\mathbb{P}\left(|\langle \mathbf{q}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon\right) \leq 2\exp\left(-\frac{2N\epsilon^2}{L^2\|\mathbf{q}\|^2}\right). \tag{4.33}$$

We now define the set $\mathcal{S}_{\boldsymbol{\alpha}} := \{\boldsymbol{\alpha}^k(t)\}_{t,k=1}^{\infty,d}$. Our next goal is to leverage (4.33) and derive a probability bound similar to (4.31) that *uniformly* holds for *all* $\mathbf{q} \in \mathcal{S}_{\boldsymbol{\alpha}}$. To this end, let

$$\mathbb{C}_\xi := \{\mathbf{c}_1, \ldots, \mathbf{c}_{d_\xi}\} \subset \Delta \quad \text{s.t.} \quad \mathcal{S}_{\boldsymbol{\alpha}} \subseteq \bigcup_{q=1}^{d_\xi} B(\mathbf{c}_q, \xi) \tag{4.34}$$

denote an ξ-covering of $\mathcal{S}_{\boldsymbol{\alpha}}$ in terms of the ℓ_2 norm and define $\bar{\mathbf{c}} := \arg\max_{\mathbf{c}\in\mathbb{C}_\xi} \|\mathbf{c}\|$. It then follows from (4.33) and the union bound that

$$\mathbb{P}\left(\sup_{\mathbf{c}\in\mathbb{C}_\xi} |\langle \mathbf{c}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon\right) \leq 2d_\xi \exp\left(-\frac{2N\epsilon^2}{L^2\|\bar{\mathbf{c}}\|^2}\right). \tag{4.35}$$

In addition, we have

$$\sup_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}}} |\langle \mathbf{q}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \overset{(a)}{\leq} \sup_{\mathbf{c}\in\mathbb{C}_\xi} |\langle \mathbf{c}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| + \sup_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}},\mathbf{c}\in\mathbb{C}_\xi} \|\mathbf{q}-\mathbf{c}\|\|\mathbf{h}(t)\|, \tag{4.36}$$

where (a) is due to triangle and Cauchy–Schwarz inequalities. Trivially, $\sup_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}},\mathbf{c}\in\mathbb{C}_\xi} \|\mathbf{q}-\mathbf{c}\| \leq \xi$ from the definition of $\mathbb{C}_\xi$, while $\|\mathbf{h}(t)\| \leq \sqrt{|\mathcal{J}'|}L$ from the definition of $\mathbf{h}(t)$ and Assumption 3. Combining (4.35) and (4.36), we get

$$\mathbb{P}\left(\sup_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}}} |\langle \mathbf{q}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon + \sqrt{|\mathcal{J}'|}\xi L\right) \leq 2d_\xi \exp\left(-\frac{2N\epsilon^2}{L^2\|\bar{\mathbf{c}}\|^2}\right). \tag{4.37}$$

We now define $\bar{\boldsymbol{\alpha}} := \arg\max_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}}} \|\mathbf{q}\|$. It can then be shown from the definitions of $\mathbb{C}_\xi$ and $\bar{\mathbf{c}}$ that

$$\|\bar{\mathbf{c}}\|^2 \leq 2(\|\bar{\boldsymbol{\alpha}}\|^2 + \xi^2). \tag{4.38}$$

Therefore, picking any $\epsilon' > 0$, and defining $\epsilon := \epsilon'/2$ and $\xi := \epsilon'/(2L\sqrt{|\mathcal{J}'|})$, we have from (4.37) and (4.38) that

$$\mathbb{P}\left(\sup_{\mathbf{q}\in\mathcal{S}_{\boldsymbol{\alpha}}} |\langle \mathbf{q}, \mathbf{h}(t)\rangle - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon'\right) \leq 2d_\xi \exp\left(-\frac{4|\mathcal{J}'|N\epsilon'^2}{4L^2|\mathcal{J}'|\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon'^2}\right). \tag{4.39}$$

Note that (4.39) is derived for one dimension. We then have for all dimensions that fixing any $\mathbf{v}(t)$,

$$\mathbb{P}\left(\|\mathbf{g}_2(t) - \mathbb{E}[\nabla f(\mathbf{v}(t))]\| \geq \sqrt{d}\epsilon'\right) \leq 2d_\xi \exp\left(-\frac{4|\mathcal{J}'|N\epsilon'^2}{4L^2|\mathcal{J}'|\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon'^2}\right). \tag{4.40}$$

In order to obtain the desired uniform bound, we next need to remove the dependence on $\mathbf{v}(t)$ in (4.40). Here we write $\mathbf{g}_2(t)$ as $\mathbf{g}_2(\mathbf{v}(t))$ to show the dependency of $\mathbf{g}_2$ on $\mathbf{v}(t)$. We first claim that $\mathbf{v}(t) \in \mathbb{W} := \{\mathbf{v} : \|\mathbf{v}\|_\infty \leq \Gamma\}$ for some Γ and all t. We will verify this claim later in the analysis. We then define $\mathbf{E}_\zeta := \{\mathbf{e}_1, \ldots, \mathbf{e}_{m_\zeta}\} \subset \mathbb{W}$ to be a ζ-covering of $\mathbb{W}$ in terms of the ℓ_2 norm. It then follows from (4.40) that

$$\mathbb{P}\left(\sup_{\mathbf{e}\in\mathbb{E}_\zeta} \|\mathbf{g}_2(\mathbf{e}) - \mathbb{E}[\nabla f(\mathbf{e})]\| \geq \sqrt{d}\epsilon'\right) \leq 2d_\xi m_\zeta \exp\left(-\frac{4|\mathcal{J}'|N\epsilon'^2}{4L^2|\mathcal{J}'|\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon'^2}\right). \quad (4.41)$$

Similar to (4.36),we can also write

$$\begin{aligned}\sup_{\mathbf{v}\in\mathbb{W}} \|\mathbf{g}_2(\mathbf{v}) - \mathbb{E}[\nabla f(\mathbf{v})]\| &\leq \sup_{\mathbf{e}\in\mathbb{E}_\zeta} \|\mathbf{g}_2(\mathbf{e}) - \mathbb{E}[\nabla f(\mathbf{e})]\| \\ &+ \sup_{\mathbf{e}\in\mathbb{E}_\zeta,\mathbf{v}\in\mathbb{W}} \Big[\|\mathbf{g}_2(\mathbf{v}) - \mathbf{g}_2(\mathbf{e})\| + \|\mathbb{E}[\nabla f(\mathbf{e})] - \mathbb{E}[\nabla f(\mathbf{v})]\| \Big]. \quad (4.42)\end{aligned}$$

Further, we have from Assumption 3 and definition of the set $\mathbf{E}_\zeta$ that

$$\sup_{\mathbf{e},\mathbf{v}} \|\mathbf{g}_2(\mathbf{v}) - \mathbf{g}_2(\mathbf{e})\| \leq L'\zeta, \quad \text{and} \quad (4.43)$$

$$\sup_{\mathbf{e},\mathbf{v}} \|\mathbb{E}[\nabla f(\mathbf{e})] - \mathbb{E}[\nabla f(\mathbf{v})]\| \leq L'\zeta. \quad (4.44)$$

We now fix $\epsilon'' > 0$, and define $\epsilon' := \epsilon''/2\sqrt{d}$ and $\zeta := \epsilon''/4L'$. We then obtain the following from (4.40)–(4.44):

$$\mathbb{P}\left(\sup_{\mathbf{v}\in\mathbb{W}} |\mathbf{g}_2(\mathbf{v}) - \mathbb{E}[\nabla f(\mathbf{v})]| \geq \epsilon''\right) \leq 2d_\xi m_\zeta \exp\left(-\frac{4|\mathcal{J}'|N\epsilon''^2}{16L^2|\mathcal{J}'|d\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon''^2}\right). \quad (4.45)$$

Since $\mathbf{v}(t) \in \mathbb{W}$ for all t, we then have

$$\mathbb{P}\left(\sup_{t} |\mathbf{g}_2(\mathbf{v}(t)) - \mathbb{E}[\nabla f(\mathbf{v}(t))]| \geq \epsilon''\right) \leq 2d_\xi m_\zeta \exp\left(-\frac{4|\mathcal{J}'|N\epsilon''^2}{16L^2|\mathcal{J}'|d\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon''^2}\right). \quad (4.46)$$

The proof now follows from (4.46) and the following facts about the covering numbers of the sets $\mathcal{S}_{\boldsymbol{\alpha}}$ and $\mathbb{W}$: (1) Since $\mathcal{S}_{\boldsymbol{\alpha}}$ is a subset of Δ, which can be circumscribed by a sphere in $\mathbb{R}^{|\mathcal{J}'|-1}$ of radius $\sqrt{\frac{|\mathcal{J}'|-1}{|\mathcal{J}'|}} < 1$, we can upper bound d_ξ by $\left(\frac{12L\sqrt{|\mathcal{J}'|d}}{\epsilon''}\right)^{|\mathcal{J}'|}$ [33]; and (2) Since $\mathbb{W} \subset \mathbb{R}^d$ can be circumscribed by a sphere in $\mathbb{R}^d$ of radius $\Gamma\sqrt{d}$, we can upper bound m_ζ by $\left(\frac{12L'\Gamma\sqrt{d}}{\epsilon''}\right)^d$. We then conclude that

$$\sup_{t} |\mathbf{g}_2(\mathbf{v}(t)) - \mathbb{E}[\nabla f(\mathbf{v}(t))]| < \epsilon'' \quad (4.47)$$

with probability exceeding

$$1 - 2\exp\left(-\frac{4|\mathcal{J}'|N\epsilon''^2}{16L^2|\mathcal{J}'|d\|\bar{\boldsymbol{\alpha}}\|^2 + \epsilon''^2} + |\mathcal{J}'|\log\left(\frac{12L\sqrt{|\mathcal{J}'|d}}{\epsilon''}\right) + d\log\left(\frac{12L'\Gamma\sqrt{d}}{\epsilon''}\right)\right). \tag{4.48}$$

Equivalently, by ignoring the log terms, we have with probability at least $1-\delta$,

$$\sup_t |\mathbf{g}_2(\mathbf{v}(t)) - \mathbb{E}[\nabla f(\mathbf{v}(t))]| < \mathcal{O}\left(\sqrt{\frac{4L^2 d\|\bar{\boldsymbol{\alpha}}\|^2\log\frac{2}{\delta}}{N}}\right) \tag{4.49}$$

Lemma 5 shows that $\mathbf{g}_2(t)$ converges to the gradient of statistical risk in probability. Now we focus on $a_1 = \|\mathbf{x}(t+1) - \mathbf{w}^*\|$. Note that $\mathbf{x}(t+1)$ is obtained by $\mathbf{v}(t)$ taking a regular gradient descent step of $\mathbb{E}[f(\mathbf{v}(t))]$ with step size $\rho(t)$. When Assumption 5 and 3 are satisfied, it is well understood [42, Ch.9] that the gradient descent step satisfies

$$\|\mathbf{x}(t+1) - \mathbf{w}^*\| \le \sqrt{1-\lambda\rho(t)}\|\mathbf{v}(t) - w^*\|. \tag{4.50}$$

Then we have

$$a_1(t+1) = \|\mathbf{x}(t+1) - \mathbf{w}^*\| \le \sqrt{1-\lambda\rho(t)}\|\mathbf{v}(t) - w^*\| \le a_4(t) + \sqrt{1-\lambda\rho(t)}\|\mathbf{w}_j(t) - \mathbf{w}^*\|. \tag{4.51}$$

Now we can write the property of sequence $\mathbf{w}_j$ for some $j \in \mathcal{J}'$ as

$$\|\mathbf{w}_j(t+1) - \mathbf{w}^*\| \le \sqrt{1-\lambda\rho(t)}\|\mathbf{w}_j(t) - \mathbf{w}^*\| + a_2(t) + a_3(t) + 2a_4(t). \tag{4.52}$$

It follows from (4.11), (4.25), and Lemma 5 that with probability at least δ,

$$\|\mathbf{w}_j(t+1) - \mathbf{w}^*\| \le \sqrt{1-\lambda\rho(t)}\|\mathbf{w}_j(t) - \mathbf{w}^*\| + \mathcal{O}\left(\sqrt{d}\rho(t)\right) + \mathcal{O}\left(\sqrt{\frac{\|\bar{\boldsymbol{\alpha}}\|^2\log\frac{2}{\delta}}{N}}\right). \tag{4.53}$$

When choosing $\rho(t)$ to be $\mathcal{O}(1/t)$, the second term on the right hand side of (4.53) is $\mathcal{O}(\sqrt{d}/t)$. Note that the right hand side of (4.53) converges to 1 as $t \to 0$ and $N \to 0$. We have shown $\mathbf{w}_j(t) \xrightarrow{t} \mathbf{v}(t)$ in the consensus analysis. We then show that $\mathbf{v}(t) \xrightarrow{N,t} \mathbf{w}^*$.

We define $\bar{f}(\mathbf{v}(t)) := \mathbb{E}[f(\mathbf{v}(t)]$ and establish $\bar{f}(\mathbf{v}(t)) \xrightarrow{t,N} \bar{f}(\mathbf{w}^*)$ in probability. It then follows from [35, Theorem 4.4] and our assumptions that $\mathbf{w}^*$ is a strong minimizer of $\bar{f}(\cdot)$ and, therefore, $\mathbf{v}(t) \xrightarrow{t,N} \mathbf{w}^*$ in probability.

In order to prove the aforementioned claim, we fix any $\epsilon > 0$ and show that $\bar{f}(\mathbf{v}(t)) - \bar{f}(\mathbf{w}^*) \leq \epsilon$ for all large enough t with probability that approaches 1 as $N \to \infty$. To this end, we claim that $\bar{f}(\mathbf{v}(t))$ for all t greater than some $t' \in \mathbb{N}$ is a strictly monotonically decreasing function with probability 1 as long as $\|\nabla \bar{f}(\mathbf{v}(t))\| > \epsilon_\nabla$ for $\epsilon_\nabla := \frac{\epsilon}{3\Gamma}$. This claim means that $\|\nabla \bar{f}(\mathbf{v}(t))\|$ eventually becomes smaller than ϵ_∇ with probability 1 for large enough t, which implies

$$\begin{aligned} \bar{f}(\mathbf{v}(t)) - \bar{f}(\mathbf{w}^*) &\leq -\langle \nabla \bar{f}(\mathbf{v}(t)), (\mathbf{w}^* - \mathbf{v}(t)) \rangle \\ &\leq \|\nabla \bar{f}(\mathbf{v}(t))\| (\|\mathbf{w}^*\|_2 + \|\mathbf{v}(t)\|) \\ &\leq 2\epsilon_\nabla \Gamma < \epsilon \end{aligned} \tag{4.54}$$

because of convexity of $\bar{f}(\cdot)$, the Cauchy–Schwarz inequality, and our assumptions. Since this is the desired result, we need only focus on the claim of strict monotonicity of $\bar{f}(\mathbf{v}(t))$ for this lemma. To prove this claim, note from Assumption 3 that

$$\bar{f}(\mathbf{v}(t+1)) \leq \bar{f}(\mathbf{v}(t)) + \langle \nabla \bar{f}(\mathbf{v}(t)), (\mathbf{v}(t+1) - \mathbf{v}(t)) \rangle + \frac{L}{2} \|\mathbf{v}(t+1) - \mathbf{v}(t)\|^2. \tag{4.55}$$

Next, we can write the update of $\mathbf{v}(t)$ as follows:

$$\mathbf{v}(t+1) = \mathbf{v}(t) - \rho(t)\mathbf{g}_2(t) + \mathbf{d}(t).$$

where $\mathbf{d}(t) = \rho(t)\mathbf{g}_1(t) - \rho(t)\mathbf{g}_2(t)$. Plugging this into (4.55) results in

$$\bar{f}(\mathbf{v}(t)) - \bar{f}(\mathbf{v}(t+1)) \geq -\langle \nabla \bar{f}(\mathbf{v}(t)), \mathbf{d}(t) \rangle + \rho(t) \langle \mathbf{g}_2(t), \nabla \bar{f}(\mathbf{v}(t)) \rangle - \frac{L}{2} \|\mathbf{d}(t) - \rho(t)\mathbf{g}_2(t)\|^2. \tag{4.56}$$

The right-hand side of (4.56) strictly lower bounded by 0 implies strict monotonicity of $\bar{f}(\mathbf{v}(t))$. Simple algebraic manipulations show that this is equivalent to the condition

$$\frac{L\rho(t)^2}{2} \|\mathbf{g}_2(t)\|^2 < -\langle \nabla \bar{f}(\mathbf{v}(t)), \mathbf{d}(t) \rangle + \rho(t) \langle \mathbf{g}_2(t), \nabla \bar{f}(\mathbf{v}(t)) \rangle + L\rho(t) \langle \mathbf{d}(t), \mathbf{g}_2(t) \rangle - \frac{L}{2} \|\mathbf{d}(t)\|^2. \tag{4.57}$$

Next, from Lemma 5 that $\mathbb{P}(\|\mathbf{g}_2(t) - \nabla \bar{f}(\mathbf{v}(t))\| \leq \epsilon')$ converges to 1 simultaneously for all t for any $\epsilon' > 0$. We therefore have with probability 1 (as $N \to \infty$) that

$$\|\mathbf{g}_2(t)\|^2 \leq \|\nabla \bar{f}(\mathbf{v}(t))\|^2 + \epsilon'^2 + 2\epsilon' \|[\nabla \bar{f}(\mathbf{v}(t))\| \tag{4.58}$$

and

$$\langle \mathbf{g}_2(t), \nabla \bar{f}(\mathbf{v}(t))\rangle \geq \|\nabla \bar{f}(\mathbf{v}(t))\|^2 - \epsilon' \|\nabla \bar{f}(\mathbf{v}(t))\|. \tag{4.59}$$

The inequalities (4.58) and (4.59) make the following condition sufficient for (4.57) to hold with high probability:

$$\begin{aligned}\frac{L\rho(t)^2}{2}\|\nabla \bar{f}(\mathbf{v}(t))\|^2 + \frac{L\rho(t)^2}{2}\epsilon'^2 + L\rho(t)^2\epsilon'\|\nabla \bar{f}(\mathbf{v}(t))\| &< \rho(t)\|\nabla \bar{f}(\mathbf{v}(t))\|^2 - \rho(t)\epsilon'\|\nabla \bar{f}(\mathbf{v}(t))\| \\ &+L\rho(t)\langle \mathbf{d}(t), \mathbf{g}_2(t)\rangle - \langle \nabla \bar{f}(\mathbf{v}(t)), \mathbf{d}(t)\rangle - \frac{L}{2}\|\mathbf{d}(t)\|^2.\end{aligned} \tag{4.60}$$

Noting that $\frac{L\rho(t)^2}{2} - \rho(t) < 0$, we can rewrite (4.60) as

$$\begin{aligned}\|\nabla \bar{f}(\mathbf{v}(t))\| > \Bigg(&\frac{2\epsilon'\|\nabla \bar{f}(\mathbf{v}(t))\|}{2 - L\rho(t)} + \frac{2\langle \nabla \bar{f}(\mathbf{v}(t)), \mathbf{d}(t)\rangle}{2\rho(t) - L\rho(t)^2} - \frac{2L\langle \mathbf{d}(t), \mathbf{g}_2(t)\rangle}{2 - L\rho(t)} \\ &+ \frac{L\|\mathbf{d}(t)\|^2}{2\rho(t) - L\rho(t)^2} + \frac{2L\rho(t)\epsilon'\|\nabla \bar{f}(\mathbf{v}(t))\|}{2 - L\rho(t)} + \frac{L\rho(t)\epsilon'^2}{2 - L\rho(t)}\Bigg)^{\frac{1}{2}}.\end{aligned} \tag{4.61}$$

Condition (4.61) is sufficient for (4.57) to hold in probability. We now note that $\|\nabla \bar{f}(\cdot)\|$ and $\|\mathbf{g}_2(\cdot)\|$ in (4.61) can be upper bounded by some constants $L_{\bar{\nabla}}$ and L_{∇} by virtue of Assumption 3 and the definition of $\mathbb{W}$. Together with Cauchy–Schwarz inequality, we have the following sufficient condition for (4.57):

$$\|\nabla \bar{f}(\mathbf{v}(t))\| > \left(\frac{2L_{\bar{\nabla}}\epsilon' + 2LL_{\bar{\nabla}}\rho(t)\epsilon' + L\rho(t)\epsilon'^2}{2 - L\rho(t)} + \frac{2LL_{\nabla}\|\mathbf{d}(t)\|}{2 - L\rho(t)} + \frac{2L_{\bar{\nabla}}\|\mathbf{d}(t)\| + L\|\mathbf{d}(t)\|^2}{2\rho(t) - L\rho(t)^2}\right)^{\frac{1}{2}}. \tag{4.62}$$

The right-hand side of (4.62) can be made arbitrarily small (and, in particular, equal to ϵ_{∇}) through appropriate choice of ϵ' and large enough t; indeed, we have from our assumptions, Lemma 5, and the definitions of $\rho(t)$ and $\mathbf{d}(t)$ that both $\mathbf{d}(t)$ and $\mathbf{d}(t)/\rho(t)$ converge to 0 as $t \to \infty$.

This completes the proof, except that we need to validate one remaining claim. The claim is that $\bar{f}(\mathbf{v}(t))$ cannot converge when $\|\nabla \bar{f}(\mathbf{v}(t))\| > \epsilon_{\nabla}$. We prove this by contradiction. Suppose $\exists \epsilon_0 > 0$ such that $\|\nabla \bar{f}(\mathbf{v}(t))\| - \epsilon_{\nabla} > \epsilon_0$ for all t. We know $\exists t' \in \mathbb{N}$ such that the right hand side of (4.62) becomes smaller than ϵ_{∇} for all $t \geq t'$. Therefore, adding ϵ_0 to the right hand side of (4.62) and combining with (4.56) gives

$\forall t \geq t'$:

$$\bar{f}(\mathbf{v}(t)) - \bar{f}(\mathbf{v}(t+1)) \geq (2\rho(t) - L\rho(t)^2)(\epsilon_0^2 + 2\epsilon_\nabla \epsilon_0). \tag{4.63}$$

Taking summation on both sides of (4.63) from $t = t'$ to ∞, and noting that $\sum_{t=t'}^{\infty} \rho(t) = \infty$ and $\sum_{t=t'}^{\infty} \rho(t)^2 < \infty$, gives us $\bar{f}(\mathbf{v}(t')) - \lim_{t\to\infty} \bar{f}(\mathbf{v}(t)) \geq \infty$. This contradicts the fact that $\bar{f}(\cdot)$ is lower bounded, thereby validating our claim.

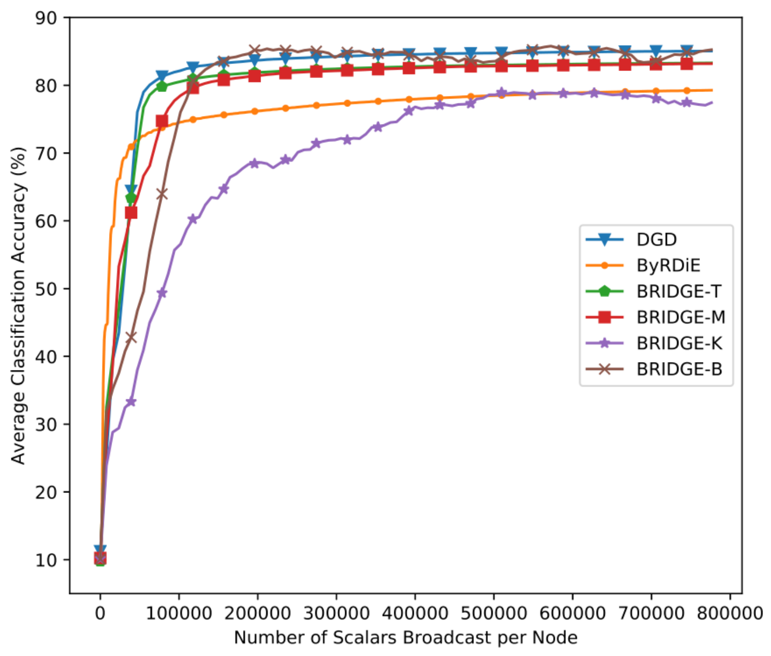
Now we take advantage of the monotone result to prove the claim we made earlier that $\mathbf{v}(t) \in \mathbb{W}$ for all t. Since (4.62) can be satisfied after some t', we define a constant $C_0 = \max_{t \leq t'} \bar{f}(\mathbf{v}(t))$ and a set $\mathbb{W}_0 := \{\mathbf{v} : \bar{f}(\mathbf{v}) \leq C_0\}$. Then define a set $\mathbb{W} := \{\mathbf{v} : \|\mathbf{v}\|_\infty \leq \Gamma\}$ where we can always find a Γ satisfying $\mathbb{W}_0 \subset \mathbb{W}$. We then show that $\mathbf{v}(t) \in \mathbb{W}$ for all t. It follows trivially that $\mathbf{v}(t) \in \mathbb{W}_0 \subset \mathbb{W}$ for $t \leq t'$. Since (4.62) is satisfied and $\mathbf{v}(t') \in \mathbb{W}$, with probability 1, $\bar{f}(\mathbf{v}(t'+1)) < \bar{f}(\mathbf{v}(t'))$. Thus, $\mathbf{v}(t'+1) \in \mathbb{W}$. Then the claim can be proven by induction.

Given that $\mathbf{w}_j(t) \to \mathbf{w}^*$, inequality (4.53) shows a sublinear convergence rate. The analysis of Theorem 8 is complete.

4.3 Numerical analysis

The decentralized system in our experimental setup involves a total of $M = 20$ nodes in the network, with a communications link (edge) between two nodes decided by a random coin flip. Once a random topology is generated, we ensure each node has at least $4b + 1$ nodes in its neighborhood (a condition imposed due to BRIDGE-B screening). The training data at each node corresponds to $N = 2,000$ samples randomly selected from the MNIST dataset. The performance of each method is reported in terms of classification accuracy, averaged over $(M - b)$ nodes and a total of 10 independent Monte Carlo trials, as a function of the number of scalars broadcast per node. The final results, shown in Figure 4.1, correspond to two sets of experiments: (i) the faultless setting in which none of the nodes actually behaves maliciously; and (ii) the setting in which two of the 20 nodes are Byzantine, with each Byzantine node broadcasting every coordinate of the iterate as a uniform random variable between -1 and 0. Note that this Byzantine attack strategy is by no means the most potent in all decentralized settings. However, this particular strategy has been selected after careful evaluation of the impact of different strategies proposed in works such as [43–45] on our particular experimental setup. Finally, with the exception of DGD, all methods are initialized with parameter $b = 2$ in both faultless and faulty scenarios.

It can be seen from Figure 4.1 that, other than ByRDiE and BRIDGE-K, all methods perform almost as well as DGD in the faultless case. In the presence of Byzantine nodes, however, DGD completely falls apart, whereas the performances of all resilient algorithms are still comparable with a minor decrease. In both cases, however, BRIDGE-B is quite effective, except that it has stringent topology requirements. We conclude by pointing out that the algorithms under BRIDGE framework are mostly (except for BRIDGE-K) more efficient than ByRDiE because of its gradient-descent-based nature.



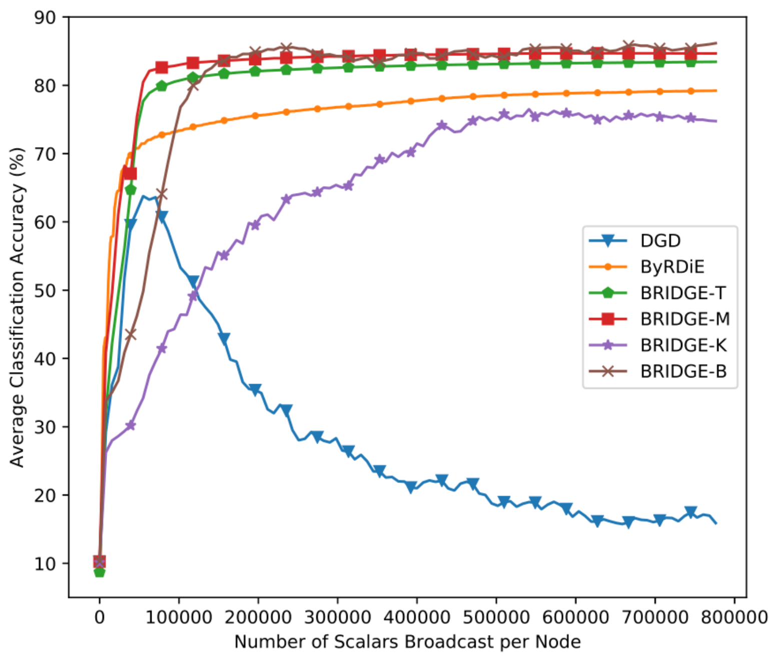


Figure 4.1: Performance comparison of different decentralized learning methods in both faultless and Byzantine settings. Byzantine-resilient algorithms in both settings operate under the assumption of $b = 2$.

4.4 Concluding remarks

In this chapter, we have introduced a Byzantine-resilient decentralized algorithm named BRIDGE. The gradient-descent-based nature makes the algorithm suitable for learning tasks in high dimension spaces. Based on the general idea of adding a screening step before each node updates its local variable, a number of different screening methods can be adopted by the framework. In the future, there is a great interest to combine more screening methods with the framework to improve its robustness or efficiency.

Chapter 5

Conclusion

In this dissertation, we have discussed different methods to complete decentralized machine learning tasks in the presence of Byzantine failures. The robustness against Byzantine failure is a relatively new topic, in both distributed and decentralized settings. Despite recent advances, Byzantine-resilient inference and learning remain an active area of research with several open problems. Much of the focus in distributed inference has been on the somewhat restrictive model in which Byzantine nodes do not collude. Collaborative Byzantine attacks, on the other hand, can be much more potent than independent ones. A fundamental understanding of mechanisms for safeguarding against such attacks remains a relatively open problem in distributed inference. Byzantine-resilient distributed estimation under nonlinear models is another problem that has been relatively unexplored. In the case of Byzantine-resilient distributed learning, existing works have only scratched the surface. Convergence and/or learning rates of many of the proposed methods remain unknown. In addition, while stochastic gradient descent [46] is a workhorse of machine learning, approaches such as accelerated first-order methods (e.g., accelerated gradient descent), first-order dual methods (e.g., ADMM), and second-order methods (e.g., Newton's method) do play important roles in machine learning. However, the resilience of distributed variants of such methods to Byzantine attacks has not been investigated in the literature.

The lack of a central server, the need for consensus, and an ad-hoc topology make it even more challenging to develop and analyze Byzantine-resilient methods for decentralized inference and learning. Much of the work in this regard is based on screening methods such as trimmed mean and median that originated in the literature on Byzantine-resilient scalar-valued consensus. This has left open the question of how other screening methods, such as the ones explored within distributed learning, might handle Byzantine attacks—both in theory and in practice—in various decentralized problems. Unlike distributed learning, any such efforts will also have to characterize the interplay between the network topology and the effectiveness of the screening procedure. The fundamental tradeoffs between the robustness and the (faultless) performance of Byzantine-resilient methods also remain largely unknown for decentralized setups. Finally, existing works on decentralized learning only guarantee sublinear convergence for strictly/strongly convex and smooth functions. Whether this can be improved by taking advantage of faster decentralized optimization frameworks or different screening methods also remains an open question.

References

[1] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47, 2002.

[2] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas. Supervised machine learning: A review of classification techniques. *Emerging Artificial Intell. Applicat. Comput. Eng.*, 160:3–24, 2007.

[3] Y. Bengio. Learning deep architectures for AI. *Found. and Trends Mach. Learning*, 2(1):1–127, 2009.

[4] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning.* MIT Press, Cambridge, MA, 2012.

[5] J. B. Predd, S. B. Kulkarni, and H. V. Poor. Distributed learning in wireless sensor networks. *IEEE Signal Process. Mag.*, 23(4):56–69, 2006.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. and Trends Mach. Learning*, 3(1):1–122, 2011.

[7] A. Nedić and A. Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Trans. Autom. Control*, 60(3):601–615, 2015.

[8] A. Mokhtari, Q. Ling, and A. Ribeiro. Network Newton distributed optimization methods. *IEEE Trans. Signal Process.*, 65(1):146–161, 2017.

[9] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Trans. Programming Languages and Syst.*, 4(3):382–401, 1982.

[10] K. Driscoll, B. Hall, H. Sivencrona, and P. Zumsteq. Byzantine fault tolerance, from theory to reality. In *Proc. Int. Conf. Computer Safety, Reliability, and Security (SAFECOMP'03)*, pages 235–248, 2003.

[11] Y. Chen, L. Su, and J. Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *Proc. ACM Measurement and Analysis of Computing Systems*, volume 1, pages 44:1–44:25, December 2017.

[12] P. Blanchard, R. Guerraoui, and J. Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proc. Advances in Neural Inf. Process. Syst. (NeurIPS'17)*, pages 118–128, 2017.

[13] C. Xie, O. Koyejo, and I. Gupta. Zeno: Byzantine-suspicious stochastic gradient descent. *arXiv preprint arXiv:1805.10032*, 2018.

[14] C. Xie, O. Koyejo, and I. Gupta. Zeno++: Robust asynchronous SGD with arbitrary number of Byzantine workers. *arXiv preprint arXiv:1903.07020*, 2019.

[15] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proc. 35th Int. Conf. Machine Learning (ICML'18)*, volume 80, pages 5650–5659, 2018.

[16] L. Li, W. Xu, T. Chen, G. Giannakis, and Q. Ling. RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *Proc. AAAI Conf. on Artificial Intelligence*, volume 33, pages 1544–1551, 2019.

[17] INC DOMO. Data never sleeps 6.0, 2019.

[18] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *Proc. Advances in Neural Information Process. Syst.*, pages 5330–5340, 2017.

[19] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, NY, second edition, 1999.

[20] A. Nedić and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Trans. Autom. Control*, 54(1):48–61, 2009.

[21] P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. *J. Mach. Learning Research*, 11:1663–1707, 2010.

[22] J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Trans. Autom. control*, 57(3):592–606, 2012.

[23] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[25] L. Su and N. Vaidya. Byzantine multi-agent optimization: Part I. *arXiv preprint arXiv:1506.04681*, 2015.

[26] H. H. Sohrab. *Basic Real Analysis*. Springer, New York, NY, second edition, 2003.

[27] S. Sundaram and B. Gharesifard. Distributed optimization under adversarial nodes. *IEEE Trans. Autom. Control*, 64(3):1063–1076, 2019.

[28] L. Su and N. H. Vaidya. Fault-tolerant multi-agent optimization: Optimal iterative distributed algorithms. In *Proc. ACM Symp. Principles of Distributed Computing*, pages 425–434, 2016.

[29] N. Vaidya. Matrix representation of iterative approximate Byzantine consensus in directed graphs. *arXiv preprint arXiv:1203.1888*, 2012.

[30] L. Su and N. Vaidya. Fault-tolerant distributed optimization (Part IV): Constrained optimization with arbitrary directed networks. *arXiv preprint arXiv:1511.01821*, 2015.

[31] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *Proc. Conf. Learning Theory (COLT'09)*, June 2009.

[32] W. Hoeffding. Probability inequalities for sums of bounded random variables. *J. American stat. assoc.*, 58(301):13–30, 1963.

[33] J. Verger-Gaugry. Covering a ball with smaller equal balls in R^n. *Discrete & Computational Geometry*, 33(1):143–155, 2005.

[34] S. J. Wright. Coordinate descent algorithms. *Math. Programming*, 151(1):3–34, 2015.

[35] C. Planiden and X. Wang. Strongly convex functions, Moreau envelopes, and the generic nature of convex functions with strong minimizers. *SIAM J. Optim.*, 26(2):1341–1364, 2016.

[36] D. Dua and E. Karra Taniskidou. UCI machine learning repository, 2017.

[37] P. J. Huber. *Robust statistics.* Springer, 2011.

[38] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram. Resilient asymptotic consensus in robust networks. *IEEE J. Sel. Areas in Commun.*, 31(4):766–781, 2013.

[39] N. H. Vaidya, L. Tseng, and G. Liang. Iterative Byzantine vector consensus in incomplete graphs. In *Proc. 15th Int. Conf. Distributed Computing and Networking*, pages 14–28, 2014.

[40] D. Alistarh, Z. Allen-Zhu, and J. Li. Byzantine stochastic gradient descent. In *Proc. Advances in Neural Inf. Process. Syst. (NeurIPS'18)*, pages 4618–4628, 2018.

[41] E. El-Mhamdi, R. Guerraoui, and S. Rouault. The hidden vulnerability of distributed learning in Byzantium. In *Proc. 35th Int. Conf. Machine Learning (ICML'18)*, volume 80, pages 3521–3530, 2018.

[42] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[43] C. Xie, S. Koyejo, and I. Gupta. Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation. *arXiv preprint arXiv:1903.03936*, 2019.

[44] M. Fang, X. Cao, J. Jia, and N. Z. Gong. Local model poisoning attacks to Byzantine-robust federated learning. *arXiv preprint arXiv:1911.11815*, 2019.

[45] G. Baruch, M. Baruch, and Y. Goldberg. A little is enough: Circumventing defenses for distributed learning. In *Proc. Advances in Neural Inf. Process. Syst. (NeurIPS'19)*, pages 8632–8642, 2019.

[46] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proc. COMPSTAT'2010*, pages 177–186. Springer, 2010.